Swarm and Evolutionary Computation 39 (2017) 323–338

Contents lists available at ScienceDirect

Swarm and Evolutionary Computation

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

A semi-supervised Genetic Programming method for dealing with noisy labels and hidden overfitting

Sara Silva[a,b,*], Leonardo Vanneschi[c], Ana I.R. Cabral[d], Maria J. Vasconcelos[e]

[a] BioISI – BioSystems & Integrative Sciences Institute, Departamento de Informática, Faculdade de Ciências, Universidade de Lisboa, 1749-016 Lisboa, Portugal
[b] CISUC, Department of Informatics Engineering, University of Coimbra, Portugal
[c] NOVA IMS, Universidade Nova de Lisboa, 1070-312 Lisboa, Portugal
[d] Department of Natural Resources, Environment and Territory, Instituto Superior de Agronomia, University of Lisbon, Tapada da Ajuda, 1349-017 Lisbon, Portugal
[e] Centro de Estudos Florestais, Instituto Superior de Agronomia, Tapada da Ajuda, 1349-017 Lisboa, Portugal

ARTICLE INFO

Keywords:
Data errors
Noisy labels
Classification
Hidden overfitting
Semi-supervised learning
Genetic Programming

ABSTRACT

Data gathered in the real world normally contains noise, either stemming from inaccurate experimental measurements or introduced by human errors. Our work deals with classification data where the attribute values were accurately measured, but the categories may have been mislabeled by the human in several sample points, resulting in unreliable training data. Genetic Programming (GP) compares favorably with the Classification and Regression Trees (CART) method, but it is still highly affected by these errors. Despite consistently achieving high accuracy in both training and test sets, many classification errors are found in a later validation phase, revealing a previously hidden overfitting to the erroneous data. Furthermore, the evolved models frequently output raw values that are far from the expected range. To improve the behavior of the evolved models, we extend the original training set with additional sample points where the class label is unknown, and devise a simple way for GP to use this additional information and learn in a semi-supervised manner. The results are surprisingly good. In the presence of the exact same mislabeling errors, the additional unlabeled data allowed GP to evolve models that achieved high accuracy also in the validation phase. This is a brand new approach to semi-supervised learning that opens an array of possibilities for making the most of the abundance of unlabeled data available today, in a simple and inexpensive way.

1. Introduction

This article tells a story. This story takes place in the realm of satellite imagery. It is a story of classification methods yielding unusually bad results, the search for the causes of such odd behavior, the discovery of human errors in the labeling of the data, and finally, the development of a method to overcome them. Why not simply eliminating the errors and redoing the work, in order to achieve the typical good results on this kind of application? Because noisy labels are very common [1–4] and usually go unnoticed, as the results seldom reveal, or even suggest, that something is wrong with the data. And even when they do, it may not be viable to go back and clean the data, and repeat the whole process. So we have to assume the data contains errors, and we have to develop learning methods that can still provide useful and reliable models under these conditions. One can state that Genetic Programming (GP) [5,6] is one of the most resilient learning methods, able to cope with noisy and faulty data, and still provide good results. But as the story will tell, even GP can be highly deceived by a very small percentage of data mislabeling.

The next section is dedicated to reviewing previous and related work on the subjects of data errors and semi-supervised learning. Section 3 describes the problem tackled and the data used in this study, including a description of the errors. Section 4 describes the workings and parameterizations of the two methods used in the beginning of our work, Classification and Regression Trees, and Genetic Programming, while Section 5 specifies the procedures used to assess their performance. Section 6 introduces the new semi-supervised GP method, explaining the differences to standard GP, and Section 7 reports all the results obtained with all the methods. Section 8 discusses these results at length, exploring the reasons for the success of the semi-supervised GP method. Finally, Section 9 summarizes the contributions of this work, and raises many additional related questions.

[*] Corresponding author. BioISI – BioSystems & Integrative Sciences Institute, Departamento de Informática, Faculdade de Ciências, Universidade de Lisboa, 1749-016 Lisboa, Portugal.
E-mail address: sara@fc.ul.pt (S. Silva).

https://doi.org/10.1016/j.swevo.2017.11.003
Received 6 April 2017; Received in revised form 29 September 2017; Accepted 12 November 2017
Available online 15 November 2017

2. Previous and related work

This section reviews the literature related to both themes addressed by this work: data errors and semi-supervised learning. Inside each theme we begin by addressing work published in the context of the wide machine learning field, followed by work in the context of Evolutionary Algorithms (EAs) and more specifically GP, and finally work related to remote sensing. We do not attempt at performing an exhaustive review of all the work published on such wide research themes, but instead we overview the amount and type of work that has been done in different specific themes, in particular the ones more related to our own work, providing pointers to more thorough surveys whenever possible.

2.1. Data errors

The objective of many learning systems is to construct a model of the world which is completely consistent with observations, based on the assumption that the data available is error-free [7]. However, this is seldom the case. According to [7] the many sources of errors may be external or internal. External errors are objective, like random errors (normally called noise) and systematic errors. Random errors are introduced by the inherent unpredictability of the world being observed, or during the transmission of the observations to the learning system. Systematic errors are more predictable, arising for instance from a problem in the device collecting data, like an instrument that is poorly calibrated. Internal errors are subjective and depend mostly on the interpretation of the data. Transversal to this classification is the concept of outlier, *i.e.*, an observation that appears to deviate markedly from other observations in a sample. The importance of outliers in statistical data and machine learning can be inferred by the very large amount of literature dealing with the subject. Interesting surveys can be found in [8] and [9].

Strategies for learning with imperfect data can focus on data cleansing, *i.e.*, identifying and repairing the errors, or on developing and using learning systems that are able to cope with them. Data mining with noisy data is considered in [10], where the authors survey other related works and propose their own error-aware method based on using noise knowledge to rectify the model built from corrupted data. According to this work, data cleansing is a limited procedure that can only be applied to certain error types from certain data sources, may lead to information loss, and constitutes in itself a potential source of additional errors.

Nevertheless, data cleansing has played a critical role in ensuring data quality, particularly with the advent of big data, where errors in data are extremely frequent. Many data cleansing algorithms have been translated into tools to detect and to possibly repair certain classes of errors such as outliers, duplicates, missing values, and violations of integrity constraints [11]. In [12], various views of data cleansing were surveyed and reviewed and a brief overview of existing data cleansing tools was given. A general framework of the data cleansing process was presented, as well as a set of general methods that can be used to address the problem. Other works followed the same path, like [13,14]. Some methods were specifically developed for big data, like [15,16]. Since different types of errors may coexist in the same data set, it is often appropriate to run more than one kind of tool. In [11], a systematic analysis of the existing data cleansing tools was performed, aimed at understanding whether these tools are robust enough to capture most errors in real-world data sets and what is the best strategy to run multiple tools to optimize the error detection effort.

Oblivious to all the efforts in cleaning data, and the problems that erroneous data may cause to learning systems, many machine learning methods are in fact equipped to perform reasonably well in modeling data with inaccuracies, as they rely on soft computing techniques to produce inexact but robust solutions. Artificial Neural Networks (ANNs), Support Vector Machines (SVMs) and Genetic Programming (GP) are some of them. In classification problems, these methods can deal not only with errors in the features, but also with errors in the labels, precisely the type addressed in our work.

An excellent review of different types of label noise and their consequences, as well as different algorithms that consider label noise, was published some years ago [17]. Among the large body of work that is reviewed, semi-supervised learning appears as one of the main noise-tolerant approaches, and a number of works on remote sensing are among the target applications. Other work not covered in this review deals with noisy labels in image annotation [18], data factorization [19], labelling pixels in aereal images [20], multiple kernel learning [21] and sentiment detection in Twitter [22].

In [23] a theoretical study on risk minimization bounds is performed on the problem of binary classification in the presence of erroneous labels, and the results are applied in developing noise-tolerant versions of SVM and weighted logistic regression. Other applied theoretical works are presented in [24,25], where the authors develop and analyse an improved logistic regression classifier that is robust to label noise. More recently, [26] studies the conditions in which a consistent classification is possible with label noise, [27] studies the use of importance reweighting to achieve an optimal classifier in the presence of noisy labels, and [28] shows that loss factorization can be directly applied on learning with poorly labeled data.

Among the most recent work, a few studies deal with the identification and correction of noisy labels. In [29] a novel L_1-optimization based sparse learning model is used to explicitly detect noisy labels, while [30] does it via a mutual consistency check using a Parzen window classifier. In [31] the unreliable labels are improved using a text label refinement algorithm, while in [32] the noisy labels are recovered as the classifier is built, using a Least-Squares SVM. In [33], the approach of repeated labeling is used in order to improve label quality, including a selective approach based on both labeling and model uncertainty.

A large and diverse body of work has also been published in the past few years focusing on using noisy labels in such varied applications as the detection of malicious network traffic [34], classification of historical notary acts [35], and time-series segmentation [36].

Noisy labels are also tackled with deep learning approaches. The notion of consistency is used in [37] to improve the predictions of a deep ANN when the labeling is missing or is subjective. Deep learning is also used in other works like [38–40]. A number of approaches rely on active learning techniques [41–44].

Compared to the huge effort that was dedicated to the detection and repairing of data errors by the larger machine learning community, the amount of work involving EAs, in particular GP, for these tasks is rather limited. Indeed, to the best of our knowledge, no paper specifically dealing with GP has ever tackled these issues directly. On the other hand, it is quite a common trend to use GP as a feature extraction process and, among the several advantages of this approach, it is typical to show that GP is resistent to data errors, and is often able to generate features that are more robust, more insightful and less prone to errors than the ones contained in the original data. The quality of a set of features can be quantified by using a machine learning method to generate a data model based on those features (and thus the fitness of the evolved features is given by the performance of this method), or by using other criteria that do not depend on any machine learning method. For instance, in [45] a measure based on information gain was employed as fitness function.

Another trend is to incorporate techniques into GP that improve its generalization ability. This was done in [46], where symbolic regression problems were solved by using new measures of fitness based on statistical learning theory, like for instance Akaike Information Criterium, Bayesian Information Criterium and Structural Risk Minimization, based on the Vapnik-Chervonenkis (VC) theory. The authors show the advantages of this type of approach and a better ability of GP to deal with noisy data.

Finally, the GP community recently focused on the relationship between overfitting, the size of the individuals and their functional complexity. An observation that is common to several contributions, *e.g.*, [47–49], is that the functional complexity of the solutions has a much clearer impact on overfitting than the size of the individuals. The fact that functionally simpler solutions have better generalization ability than complex ones could be a direct consequence of the higher robustness of simple solutions to errors in data, as discussed in [49]. Under this perspective, fostering the survival of simple solutions in GP populations by integrating measures of functional complexity in the fitness function, like the ones defined in [48], can be a method to implicitly deal with errors in data. Indeed, the GP community tends to ignore the available data cleansing tools, instead focusing on methods that can implicitly deal with the errors.

In the remote sensing community there is also a wide recognition of the problem of errors in the data and their effect on the accuracy of land cover classifications (*e.g.*, [1–4]). Most remote sensing problems involve classification, binary or multiclass, in land cover or land use classes. Remote sensing data is normally accurate, so the work published on data errors is primarily focused on mislabeling errors on the reference set, or generally speaking, label noise. The widespread usage of the term "ground truth" to denote the reference set has been criticized in [2], for implying the data are a gold standard reference. As mentioned before, remote sensing work is cited in the [17] review. For an example of recent work see [4].

2.2. Semi-supervised learning

Semi-supervised learning uses both labeled and unlabeled data to perform the learning task. This approach to learning has recently grown as a promising direction in machine learning research, because of its relevance to many practical problems where is it expensive to produce labeled data (*e.g.*, when human expertise is required) and at the same time it is easy and cheap to produce unlabeled or weak-labeled data (*e.g.*, in crowdsourcing). A rather complete overview of semi-supervised learning is offered in [50], including a brief history of semi-supervised learning, a taxonomy for semi-supervised learning methods, a detailed analysis of many of those methods, a detailed discussion on the pros and cons/risks of learning from semi-supervised data sets and the discussion of a set of benchmarks. Another wide set of benchmark data sets for semi-supervised learning can be found in [51]. On the other hand, a critical view of semi-supervised learning is given in [52].

Normally, semi-supervised learning is used on datasets with a large amount of unlabeled data and only a small quantity of labeled data. Self-labeled techniques are used for enlarging the labeled data set, by labeling previously unlabeled observations using models built on the labeled data. For a rather complete survey of self-labeled techniques, the reader is referred to [53].

Evolutionary Algorithms have often been used for semi-supervised learning, and Genetic Algorithms (GAs) are for sure the most commonly used flavor of EAs. The first contribution probably dates back to [54], where a semi-supervised clustering algorithm was proposed. The approach allowed unlabeled data with no known class to be used to improve classification accuracy. The objective function took into account both the cluster dispersion of the input attributes and a measure of cluster impurity based on the class labels. In a similar vein, some years later a novel semi-supervised clustering algorithm was proposed in [55], in which data were clustered using an unsupervised learning technique, biased towards producing clusters as pure as possible in terms of class distribution.

A different approach was presented in [56], where an EA suited to learn interpretable fuzzy if-then classification rules from partially labeled data was proposed. Interestingly, the feasibility of the approach was demonstrated also using real-world image analysis and remote sensing applications that, although different, clearly share the same nature as our own application. A few years later, an ensemble learning approach based on EAs was proposed in [57] to tackle semi-supervised learning problems, and the authors showed how the iterative nature of EAs can be, in itself, beneficial to iteratively increase the number of labeled observations. As such, this method can be seen as an evolutionary self-labeled technique. In the same year [58] GAs were used as a part of a wider machine learning system, to optimize the objective function of a standard semi-supervised SVM (S^3VM).

In [59] semi-supervised learning was used to improve an interactive GA-based system. More specifically, a surrogate model built with an improved semi-supervised learning method was employed to evaluate a part of the individuals, in order to alleviate the work of the user in performing the evaluation. The effectiveness of the proposed method was assessed on the design of sunglass lenses, a typically rather complex optimization problem. In [60] a GA was applied for regression function semi-supervised learning. Based on a few labeled examples and the agreement among the views on the unlabeled examples, the error of the algorithm was optimized, striving after minimal regularized risk, reporting excellent performance on the test problems used.

Recently, the authors of [61] pointed out that traditional semi-supervised learning tentatively labels the unlabeled data on the basis of the smoothness assumption that neighboring points should have the same label. They also observe that when this assumption is violated, unlabeled points are mislabeled, injecting noise into the classifier. Therefore, they present an alternative approach called cluster-then-label (CTL), which partitions all the data points (labeled and unlabeled) into clusters and creates a classifier by using those clusters.

Also other kinds of biological and nature inspired algorithms were used for semi-supervised learning. For instance, in [62] an Artificial Immune System (AIS) was used to determine which data is better to labeled in order to get high quality data. Also, Particle Swarm Optimization (PSO) was used for semi-supervised learning in [63]. The semi-supervised PSO simultaneously used limited labeled data and large amounts of unlabeled data to find a collection of prototypes (or centroids) that were considered to precisely represent the patterns of the whole data.

Among the different types of EAs, GP was also used with success for semi-supervised learning. For instance, in [64] a semi-supervised GP system called Active Learning GP (AGP) was proposed, instantiated for the data deduplication problem (a data compression technique for eliminating duplicate copies of repeating data), and used on semi-supervised benchmarks. In [65] a semi-supervised transductive GP algorithm, called KGP, was proposed for classification. KGP is transductive (instead of inductive), *i.e.*, it requires only a training data set with labeled and unlabeled examples, which should represent the complete data domain. The effectiveness of KGP was demonstrated on a wide set of test problems. Finally, a novel evolutionary approach was proposed in [66], which can be applied to supervised, semi-supervised and unsupervised learning tasks. The method, Grammatical Evolution Machine Learning (GEML), adapts machine learning concepts from decision tree learning and clustering methods, and integrates these into a Grammatical Evolution framework. The authors state that the framework generates human readable solutions, which explain the mechanics behind the classification decisions, offering a significant advantage over existing paradigms for unsupervised and semi-supervised learning. Even though [64–66] all deal with overfitting, and we can assume that the data errors are outliers that the models should not (over)fit, to the best of our knowledge no work has been published yet that is explicitly devoted to the use of GP in a semi-supervised manner for dealing with data errors.

However, a method called Backwarding for controlling overfitting has been used in [67] that may well be the most similar to our current work, although the authors never mention erroneous labels or semi-supervised learning. During the evolutionary process, Backwarding keeps track of the best solution on the training set (as usually done), and it also keeps track of the best solution on the training set that is also the best solution on an independent validation set. In the beginning of the evolution it is expected that both best-on-training and best-on-

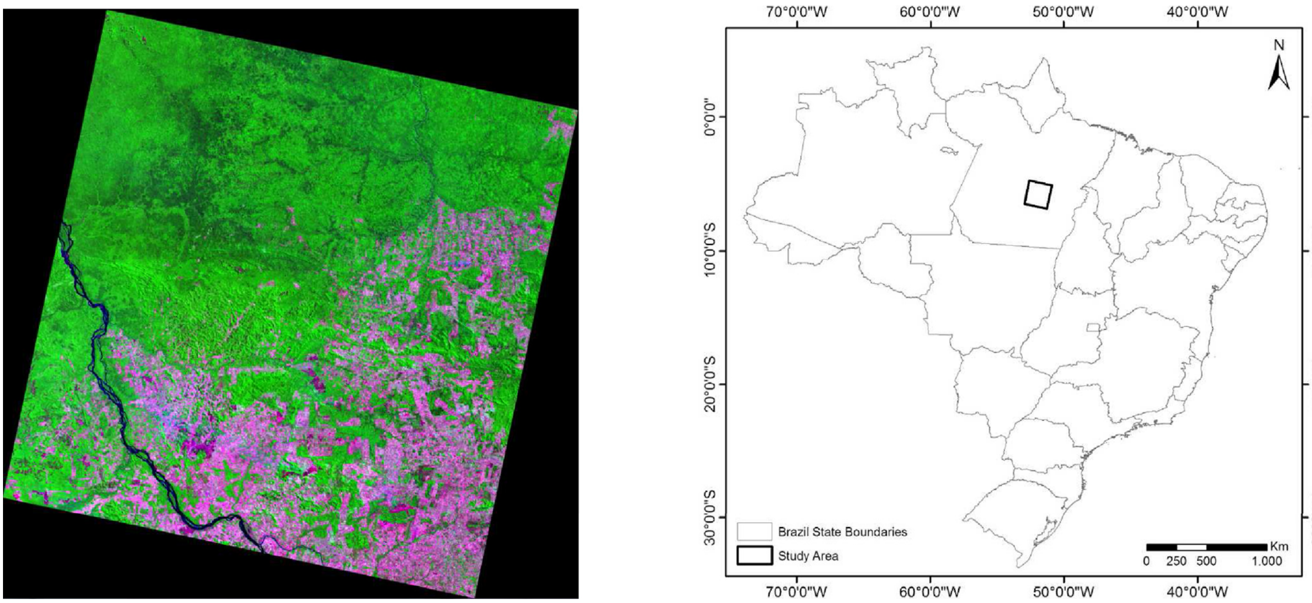

Fig. 1. Landsat 8 OLI image (left) used in this study. Map of Brazil with the location of the study area and the distribution of the administrative regions (right), where the tilted black square shows the position of the image.

validation are the same and updated frequently, but when overfitting starts occurring the best-on-validation is not updated because the fitness on the validation set does not improve. When the evolution stops (based on any stopping criteria), the algorithm returns the best-on-validation solution, not the best-on-training, thus backwarding to a generation prior to the occurrence of overfitting. This is a slightly better way of implementing early stopping (better in the sense that the search does not stop at the earliest sign of overfitting, which may be premature) and it allowed the authors to report improved results on additional unseen data in a remote sensing application. However, when classifying real images the results were still not good, and the blame was put on the lack of sufficient validation data. Therefore, they have used a much larger, unlabeled, validation set, and relaxed the definition of 'better fitness on the validation set' to simply keeping the highest number of points within the expected range of results. This led to drastically improved results, but the authors do not mention the possibility that it was the extended validation set and relaxed fitness calculation, and not the backwarding, that allowed such feat.

In remote sensing there is a vast literature about semi-supervised learning, in particular for classification tasks. A review of some methods is presented in [68]. Some methods used for dealing with noisy labels had already been mentioned in the [17] review. Among the most recent work on semi-supervised learning in remote sensing, SVM is regarded as the state of the art [69], on which many variants have been proposed (*e.g.*, [70–74]). A selection of recent non-SVM based methods include a combination of multinomial logistic regression with k-nearest neighbor [75], a co-training approach introduced by the Tracking-Learning-Detection framework [76] and, of course, deep learning [77].

3. The problem and the data

Every year, large areas of tropical savannas and woodlands burn due to natural conditions and land management practices induced by human activities. Given the high level of green house gas emissions produced by biomass burning, there is the need to define historic fire regimes so that prospective emission reduction management strategies can be well informed and their results measured, reported, and verified. Thus, it is important to develop tools for accurately and frequently mapping burned areas over large extents. Satellite data from high resolution sensors Landsat TM (Thematic Mapper), ETM+ (Enhanced Thematic Mapper Plus) and OLI (Operational Land Imager) are a valuable source of information and have been widely used in the development of automated and semi-automated methods to detect burned areas (*e.g.*, [78–81]).

In this work, a Landsat 8 OLI image over Brazil was selected, corresponding to Path/Row 225/64 and freely downloaded from GLOVIS archive of the U.S. Geological Survey (USGS) Earth Resources Observation and Science (EROS) Center.[1] It corresponds to an area located in eastern Amazonian, which lies south of the Amazon River and is drier than the central and western parts of the Amazon, with annual rainfall between 1500 mm and 2000 mm and average temperatures ranging from 23 °C to 30 °C. Forest types range from lowland Amazon forest (tall trees of up to 40 m in height) in the north through submontane dense and open forests in the south. This region is subject to frequent and extensive fires. The image was acquired on February 28, 2015, thus ensuring the presence of recently burned areas. The image contains more than 40 million pixels, where approximately 2.5 million are burned, representing an estimated total burned area of approximately 0.6%. Fig. 1 shows the image (left) and its geographical location (right).

A Landsat OLI image consists of nine different bands,[2] of which band 8 is a panchromatic band (of low spectral resolution, covering most of the visible range) and band 9 is a cirrus band (used only for cloud detection). Therefore, we have used only the first seven bands, covering Ultra Blue, Blue, Green, Red, Near Infrared (NIR) and two different Shortwave Infrared (SWIR) spectral ranges. The image was geometrically corrected to UTM-Zone 22 South, Datum WGS84 for a spatial resolution of 30 m. Visual inspection of the combination of bands 7, 5 and 4 allows depicting burned areas very clearly [82]. This is the combination used for displaying the image in Fig. 1 and also the detail in Fig. 5a.

For training and testing the methods presented here, a data set was assembled from this image. A human expert manually collected several sample points, or observations, from different burned and unburned areas. An effort was made in order to obtain a balanced data set, in order to cover not only the different non-burned land cover types, but also a large diversity of burn scars, as they do not all have the same

[1] http://glovis.usgs.gov/.

[2] https://landsat.usgs.gov/how-does-landsat-8-differ-previous-landsat-satellites.

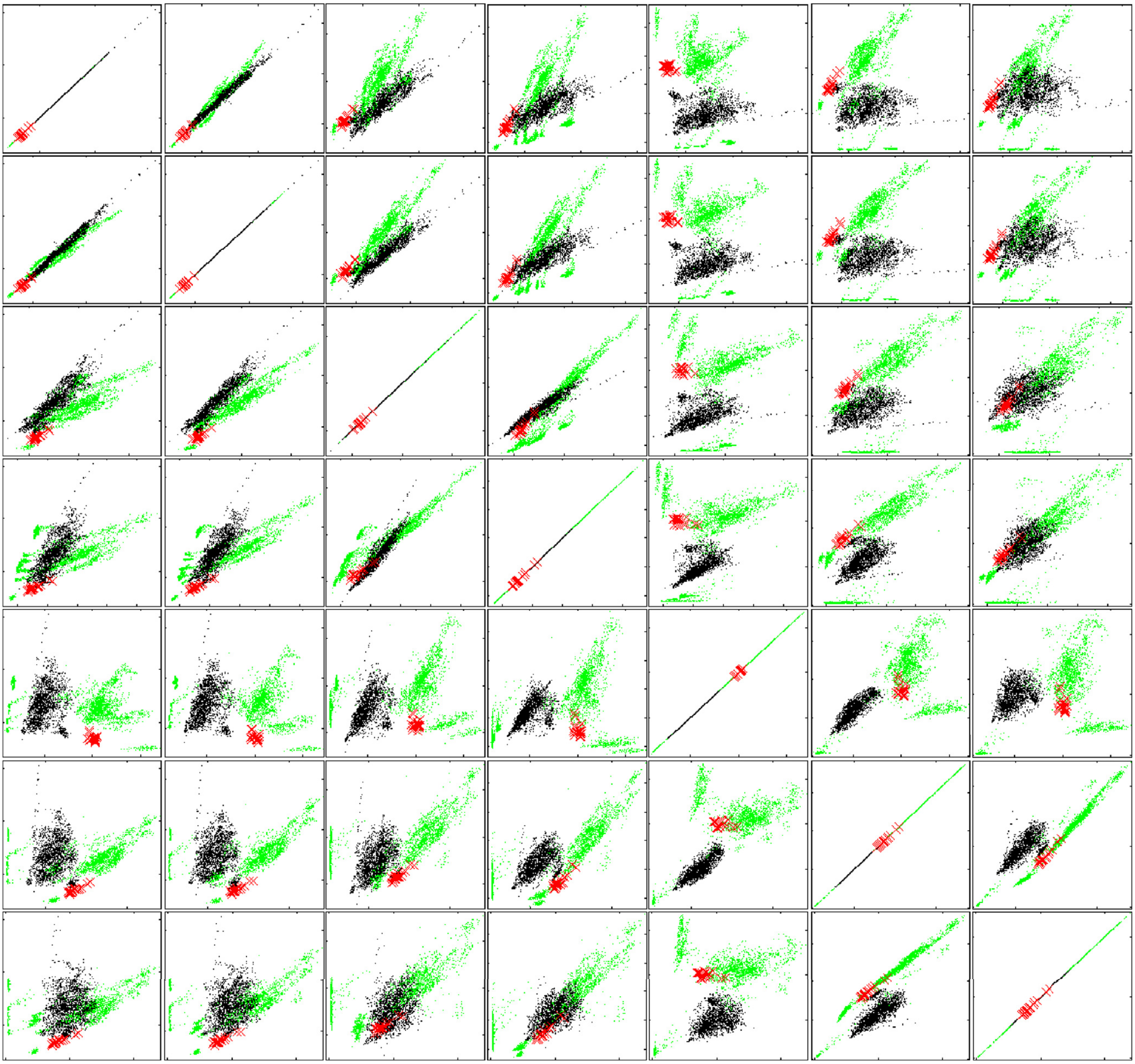

Fig. 2. Matrix of scatter plots of the reference data, each plot showing the relationship between two of the seven bands. Black dots represent burned pixels, light green represents non-burned, and the larger red markers represent the 20 erroneously labeled pixels. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

spectral characteristics. Each observation corresponds to one pixel of the image, and consists of the observed DN (Digital Number) values for the seven bands (seven explanatory variables) and the corresponding target value, or class label, 1 (burned) or 0 (unburned) (dependent variable). In total, 4872 pixels were collected and labeled by the human expert, 2053 corresponding to class 1 (burned) and 2819 corresponding to class 0 (unburned). We call this data set the reference set, or reference data.

Later it was discovered that some of the 4872 pixels were mislabeled. An area of green forest was erroneously labeled by the human expert as class 1 (burned), affecting a total of 20 pixels. This small number of pixels represents less than 0.5% of the total amount of observations in the reference set.

It is important to mention that a deeper investigation of the data revealed other problematic pixels that are considered by the human experts a potential source of confusion to the learning algorithms. In particular, one of the burned areas from which pixels where collected corresponds to understory burns which produce mixed pixels containing various degrees of green and burned signals. All the observations from this area were correctly labeled as class 1 (burned), but in another context some of them could have been labeled differently. Nonetheless, even if these problematic pixels were also to be considered erroneous (which they are not), the total amount of errors would still amount to only 1% of the reference data.

Fig. 2 shows a 7 × 7 matrix of scatter plots of the reference data, each plot showing the relationship between two of the seven bands. The pixels labeled as burned are represented in black, while the non-burned are represented in light green. The 20 erroneously labeled pixels are represented with larger red markers (×) in order to be seen. The figure brings no surprises, showing a high correlation between some of the bands, and the spectral diversity of both burned and non-burned classes. It also suggests that the task of distinguishing burned from non-

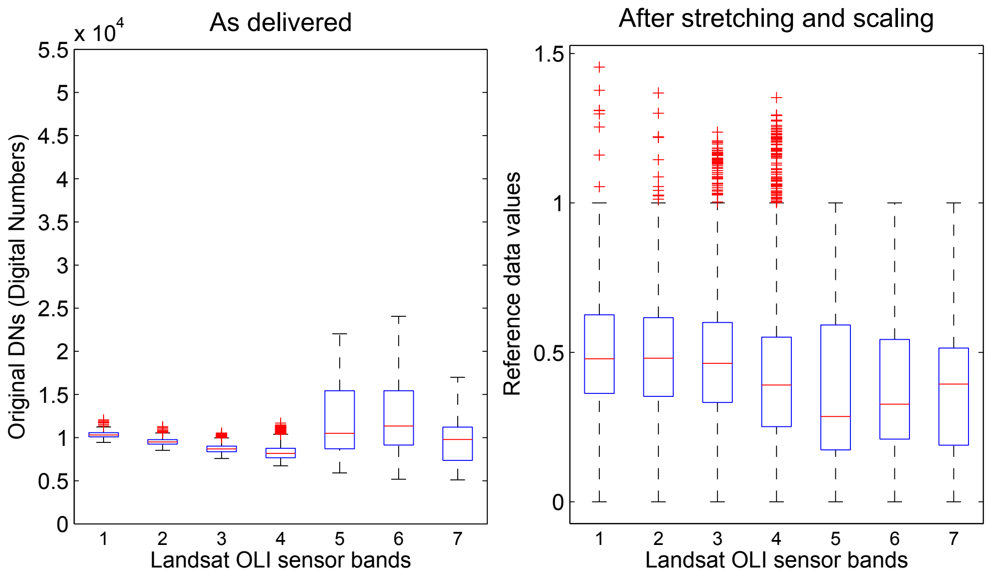

Fig. 3. Boxplots of original DNs (Digital Numbers) as delivered (left) and values obtained after stretching and scaling (right), on each of the Landsat OLI sensor bands.

burned should not be a difficult one, and reveals why the erroneous pixels can be a confounding factor for the classifiers, as these 20 points are mostly detached from the burned pixels, appearing much closer to the non-burned ones.

The Landsat OLI sensor records 12-bit values, which translates into 4096 potential grey levels for each band. However, the images are delivered as 16-bit values, scaled to 55000 grey levels.[3] Looking at the values in the reference set, one can observe that most values are concentrated on a very narrow interval, in particular for bands 1–4 (see Fig. 3, left). For this reason, the values of each band were subject to a linear stretching and scaling into the interval [0 1]. The input limits for the stretching were the minimum and maximum values of each band, excluding the outliers, which means that in the end the outliers fall outside the interval [0 1] (see Fig. 3, right).

Finally, the reference set was randomly split into training and test sets. This split was performed 30 times, each time obtaining a different partition, always using 70% of the observations for training (totaling 3411 pixels), and the remaining 30% for testing (totaling 1461 pixels). The reason for using the test set is to detect overfitting in case it occurs (more on this in Section 8.1). The reason for using 30 different partitions is to ensure that the studied methods produce consistent results across different data sets.

4. Methods - CART and GP

This section briefly describes the two methods used in the first part of the work: CART and GP. CART (Classification And Regression Trees) [83] has been a popular method in remote sensing applications for a long time now (*e.g.*, [84–87]), providing models in the form of simple interpretable rules. For example, it has been applied successfully to remote sensing data for burned area mapping in Africa (*e.g.*, [88–90]). GP is still mostly unknown outside (and sometimes even inside!) the machine learning community, despite its success in many real world applications [6,91]. Nevertheless, it is being used in a growing number of remote sensing applications (*e.g.*, [92–98]), including the identification of burned areas in satellite imagery [99].

4.1. Classification and Regression Trees

Classification trees are a non-parametric, non-linear rule based classifier that generates classification rules through an induction procedure described in [83]. They are based on a hierarchical decision scheme where the feature space is subject to a binary recursive partitioning that successively splits the data. In the Classification and Regression Tree (CART) algorithm [83], heuristic techniques are used to achieve an inverted tree type structure, starting in a root node with all the data, and generating descendent nodes with a series of splitting decisions (if-then rules) until terminals are reached.

Single tree classifiers are developed using the Gini index criteria for node splitting [83] and assuming equal class prior probabilities for burned areas. Equal classification error costs for burned and unburned classes are assumed, and terminal tree nodes are required to contain a minimum of 20 observations. Linear combinations of the variables are employed to deal more effectively with data patterns [83]. The selection of the best tree size (optimal tree), to avoid overfitting the training data, was performed using the technique cost-complexity pruning based on a test sample. This technique selects the optimal compromise between the number of tree nodes and misclassification rate, and penalizes very large trees [83]. The best tree is chosen based on three criteria: the best training accuracy, the best test accuracy and the best accuracy obtained between classes given by the variable importance. Variable importance is the sum across all nodes in the tree of the improvement scores that the predictor yields when it performs as a splitter [83].

4.2. Genetic Programming

Genetic Programming (GP) is the automated learning of computer programs, using Darwinian selection and Mendelian genetics as sources of inspiration [5,6]. Starting from an initial population of randomly created programs representing the potential solutions to a given problem, it evaluates the fitness of each, quantifying how well the program solves the problem. New generations of programs are iteratively created by selecting parents based on their fitness, and breeding them using genetic operators like crossover and mutation, where pieces of code are swapped and modified, respectively. Because fitter individuals are selected more often and given the chance to pass their best characteristics to their offspring, the population tends to improve in quality along successive generations. In this work, we use tree-based GP with standard subtree crossover and no mutation, a common setting [5]. From

[3] https://landsat.usgs.gov/landsat-8.

now on we will designate it simply as StdGP.

Regarding the remaining settings, the population is composed of 500 individuals initialized with the Ramped Half-and-Half method [5] with depths between 2 and 6, allowed to evolve for 200 generations. Selection for breeding is performed with lexicographic tournaments [100] of size 10. Selection for survival is non-elitist, meaning that each generation of offspring completely replaces the parent population. No maximum limit is imposed on the depth of the trees. Instead, bloat control is ensured by the Dynamic Operator Equalisation (DynOpEq) method [101]. The function set contains only the four basic arithmetic operators: addition, subtraction, multiplication and division, protected as in [5]. No terminals are used besides the seven variables of the problem.

We perform the classification task by evolving a regression model and then applying a cutoff to the raw output values, in order to interpret them as class labels. Therefore, the fitness function is the usual RMSE used for regression, with expected outputs 0 (not burned) and 1 (burned), and the accuracy is only calculated offline after applying a cutoff of 0.5. The best model is chosen based on training and test accuracy. The interesting thing about the raw output values of the GP models is that it can be interpreted as the level of certainty that GP has in its own classification. Outputs very close to 0 or very close to 1 denote a high certainty of the respective classes, while outputs close to 0.5 denote uncertainty between both classes. On the other hand, output values that fall very far from the [0 1] interval are undesirable, as they reveal an unstable behavior and possible asymptotes in the model.

5. Validation procedures

The process of validation used in this work includes several procedures. The goal of the work is not to obtain statistically significant results measured on the 30 data partitions, but to obtain a reliable model to perform the classification. Therefore, the first procedure is to actually choose one model among the 30 learned models, for each of the methods tested (see 4.1 and 4.2).

The second procedure is to measure the overall accuracy and Cohen's Kappa values [102] obtained by the chosen models on a large validation set. Overall accuracy is expressed as the percentage of correctly classified pixels. The Kappa coefficient is another accuracy measure, very popular with remote sensing data, as it makes some compensation for chance agreement between classes. A grid of 5000 equally spaced points is randomly placed over the image. Many of these points fall outside the image limits (the black borders in Fig. 1, right), and the remaining 3525 are interpreted and labeled by the same human experts that provided the reference set, becoming the validation set. Being randomly collected, these pixels are highly unbalanced between the two classes, with only 27 of them being labeled as class 1 (burned), representing approximately 0.8% of the entire validation set. We assume that all validation pixels are correctly labeled. None of the burned pixels of this grid overlaps with the burned pixels of the reference set.

Besides this numeric validation, a visual validation is also performed. The entire image is classified by each of the chosen models, and a human expert inspects the classifications to check if they conform to what is plausibly expected. In reality, this procedure is normally followed by another numeric validation based on comparing the automatic classifications with a manual classification where the expert delineates polygons around all the burned areas that can be visually identified. This is a lengthy global validation that was not performed in this work, as the visual inspection of the classifications immediately revealed a serious problem (Section 7).

Because of the revealed problem, a new GP method was developed (described in the next section) that partially uses the large validation set during the learning. For this reason we have built a second, much smaller validation set, from a new grid of 250 points randomly placed over the image. The 176 points not falling outside the image limits are used to validate the quality of the models a second time, again through the overall accuracy and Kappa coefficient. Four of these pixels are labeled as class 1 (burned), representing approximately 2.3% of the entire set. None of these burned pixels overlap with the burned pixels of either the reference set or the first grid. This extra validation, on top of the others, ensures that using information of the first validation set during the learning does not artificially improve the results.

6. New method - semi-supervised GP

The new method introduced here is very similar to StdGP, except for the fitness function. Truth be said, even the fitness function is very similar to the one of StdGP. While StdGP uses the RMSE as fitness, the new method also uses the RMSE as fitness. So where is the difference?! The difference is that the new method calculates de RMSE using not only the regular labeled data of the reference data set, but also an extended set of unlabeled data. We naturally call this new method Semi-Supervised GP, and from now on we will designate it simply as SSupGP.

In this work, the extended set of unlabeled data is basically what we had most readily available, i.e., the large validation set (after removing the labels). Using the validation set to calculate the fitness may seem like “cheating”, however we took special care on making sure the results are not biased by this practice (see Sections 5 and 7). Other extended sets can be used, as unlabeled data is normally abundant and easy to collect. For applications where this is not the case, it is useful to know that using the validation set without the labels is allowed.

Let us denote y_t as the output value obtained on the labeled observations t, with $t = 1..n$, and $\hat{y}_t$ the respective numeric class label (0 or 1). Let us also denote z_u as the output value obtained on the unlabeled observations u, with $u = 1..m$. The modified RMSE is calculated as

$$\sqrt{\frac{\sum_{t=1}^{n}(\hat{y}_t - y_t)^2 + \sum_{u=1}^{m}(\hat{z}_u - z_u)^2}{n+m}}, \tag{1}$$

where $\hat{z}_u$ is the class label (0 or 1) closest to the output z_u.

It is important to say that the main motivation for this modified RMSE fitness function was to improve the behavior of the models obtained by GP. In this type of application, the GP models are normally well behaved, outputting values near the interval [0 1] (for binary classification problems where the class labels are 0 and 1) both on training and on unseen data [99]. However, in this case the models exhibited an unusual behavior, returning an array of out-of-range values when asked to classify unseen data. As SSupGP was being developed in order to improve model behavior, the labeling errors were discovered. We are not sure, but these errors are probably what caused the wild behavior in the first place. What we know is that, by promoting a more constrained behavior of the models, SSupGP also proved that it can cope with such errors. Section 8 discusses the possible reasons for this.

7. Results

In this section, we first present the results obtained by both methods CART and StdGP, and then the results obtained by the new semi-supervised method SSupGP. We finish with an additional validation of the results obtained by all three methods.

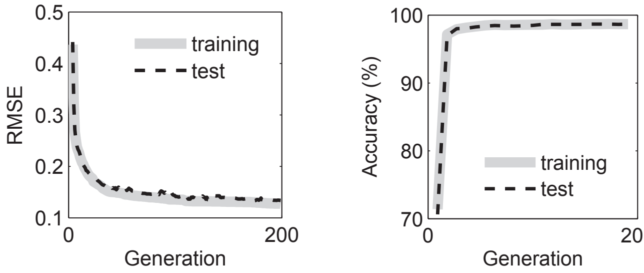

Fig. 4. Evolution of RMSE (left) and accuracy (right) on the training and test sets.

7.1. Results of CART and StdGP

The first results we show are the evolution of the fitness (measured as the RMSE between expected and predicted values, see Section 4.2) of the best individual on the training set and the fitness of the same individual on the test set during the StdGP learning, as well as the evolution of the respective overall accuracy values (Fig. 4). The lines shown in the plots report the median values calculated on the 30 runs. It can be observed that training and test RMSE values are very similar during the entire evolution, and therefore there was no reason to suspect the presence of overfitting. In terms of accuracy the values are even more similar. Also CART obtained results that did not reveal overfitting.

As described in Section 5, one model is chosen to represent each method. In both CART and StdGP all the models have very similar and very good accuracy values, both in the training sets (variance of 0.10 for CART and 0.12 for StdGP) and in the test sets (variance of 0.14 for CART and 0.15 for StdGP), and for each method they are also very similar in their form and readability. For this reason the choice is both easy and difficult, and hopefully not a very influential factor on the remainder of the work.

Table 1
Overall accuracy and Kappa values obtained by the two models on a large validation set.

Method	Accuracy (%)	Kappa
CART	95.49	0.23
StdGP	97.58	0.37

Despite the low RMSE and high accuracy values, the results obtained by both of these models on the large validation set were unexpectedly bad (Table 1). StdGP being much better than CART, it still did not reach an acceptable Kappa value (anything below 0.4 is considered as a poor classification [103]).

Following the procedures described in Section 5, each model was used to classify the entire image, and its inspection revealed the reason for the bad validation results. Both methods classified natural woodlands and regenerating patches (but not many agricultural fields) as burned areas, in particular CART. Fig. 5 shows an example, where only one large burned area should have been identified (a), and yet both CART (b) and StdGP (c) classify other large areas as burned. In the case of StdGP, in all validation procedures the classification is obtained by applying a cutoff to the raw output values of the model (as described in Section 4.2). The raw classified image, before applying the cutoff, can be seen as a greyscale representation of the certainty of the classification (d).

We found that many of the pixels wrongly classified as burned had indeed low certainty values, closer to 0.5 than to any of the class labels. However, many others, equally wrong, were showing high certainty. But the most striking observation was the wild range of raw values

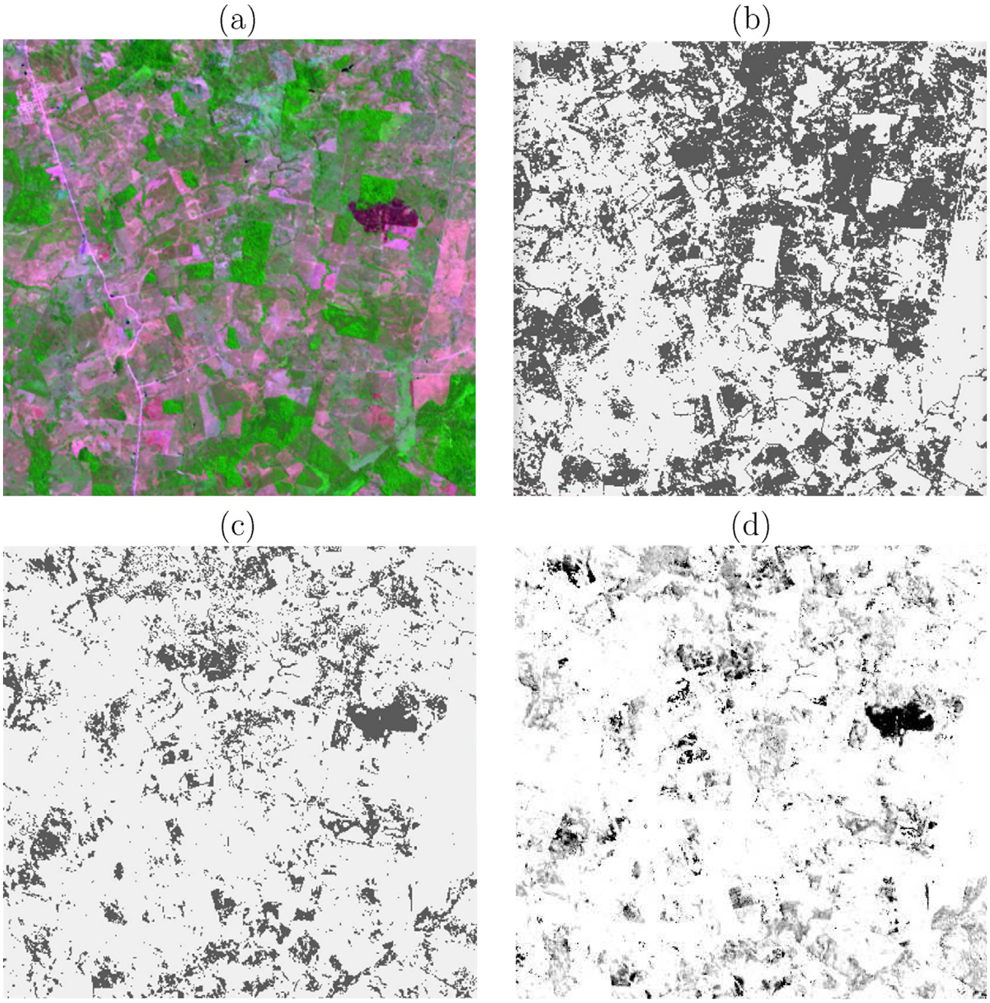

Fig. 5. Detail of the original image (a), classification by CART (b), classification by StdGP (c) and raw classified image by StdGP, before applying the cutoff (d). The burned areas appear as dark purple patches on the image (a). In the classifications (b) and (c), white represents non-burned while black represents burned. In the raw classification (d), the greyscale goes from low certainty (lighter) to high certainty (darker). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Table 2
Overall accuracy and Kappa values obtained by the three models on a large validation set.

Method	Accuracy (%)	Kappa
CART	95.49	0.23
StdGP	97.58	0.37
SSupGP	**99.43**	**0.70**

Table 3
Overall accuracy and Kappa values obtained by the three models on the extra small validation set.

Method	Accuracy (%)	Kappa
CART	94.88	0.38
StdGP	97.72	0.59
SSupGP	99.43	0.85

found throughout the entire classified image. Values very far from the interval [0 1] appear much more frequently than in other similar work we have performed in the past [99].

7.2. Results of semi-supervised SSupGP

As we were developing a method to avoid these out-of-range raw outputs (SSupGP, see Section 6), a rechecking of the reference data revealed mislabeling errors in a small percentage of observations (see Section 3). For the reasons explained earlier in the introduction, we have decided not to correct these errors. Instead, we test the performance of SSupGP in the same faulty data. We use as unlabeled data the large validation set without the class labels. The size of this validation set (3525 pixels) is approximately 70% of the size of the original reference set (4872 pixels), and more than 99% of its observations belong to class 0 (not burned).

As with the previous methods, one model was chosen among the ones evolved by SSupGP, and put through the validation procedures described in Section 5. The improvement of the results is astounding. As shown in the last row of Table 2 (we keep the previous values for comparison), on the large validation set the overall accuracy is close to 100% for SSupGP, but the main difference is the Kappa coefficient, that now is in the range 0.61–0.75 and therefore is considered to be good [103]. Furthermore, the visual inspection of the classified image revealed that the SSupGP model was not deceived by the errors in the data (see Fig. 6).

7.3. Extra validation

The visual inspection of the classified image is by itself a confirmation that SSupGP is obtaining good results overall, and not just on the pixels of the training, test, and validation sets. However, to clear any concerns regarding the admissibility of using the unlabeled validation pixels to help the learning, we perform an extra validation of the three models on the second validation set described in Section 5. The size of this extra validation set (176 pixels) is approximately 5% of the size of the first validation set (3525 pixels), and almost 98% of its observations belong to class 0 (not burned). The results are shown in Table 3 for all the methods, revealing more or less the same overall accuracy (exactly the same for SSupGP), and higher Kappa, than the ones obtained in the first validation set. According to the suggested ranges for the Kappa coefficient [103], CART produces a poor classification (<0.40) also on this set, StdGP produces a moderate classification (0.41–0.60) and SSupGP produces an almost perfect classification (>0.81). However, more important than these absolute values is the fact that the relative quality of each method, observed on the first validation set, is maintained on this extra validation set, which suggests that using the unlabeled validation data for learning does not bias the results of SSupGP.

8. Discussion

We begin this section with a discussion on the concept of overfitting, and its detection in different learning scenarios. Then we analyse the learning dynamics of both StdGP and SSupGP, in particular the effect of the labeling errors in the evolution of RMSE and accuracy, and the output values and their distance to class labels. We identify key differences between them, and finally take a closer look at the fitness function of SSupGP, discussing its implications and proposing an explanation for the success of this method.

8.1. On the concept of overfitting

Far from surveying all the different meanings and usages given to the concept of overfitting in the scientific literature, we do however provide a short briefing on how we use it in this work, and how different it may be from other works, in order to avoid misinterpretations during the discussion of the results. First of all, overfitting is a broad concept

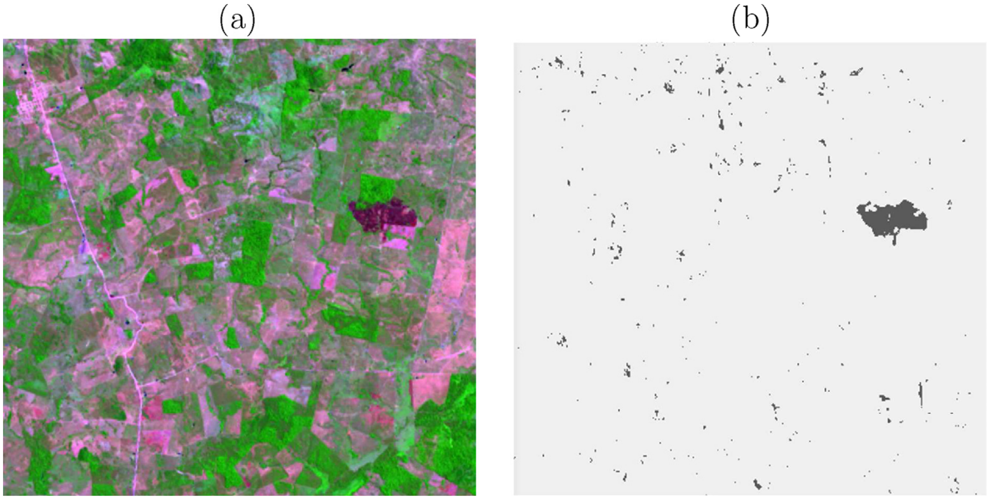

Fig. 6. Detail of the original image (a) and classification by SSupGP (b). The burned areas appear as dark purple patches on the image. In the classification, white represents non-burned while black represents burned. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Table 4
Overall accuracy and Kappa values obtained by CART and StdGP on both validation sets after learning with the error-free reference set.

Method	Large validation set		Method	Small validation set	
	Accuracy (%)	Kappa		Accuracy (%)	Kappa
CART	99.82	0.87	CART	99.43	0.85
StdGP	99.77	0.85	StdGP	99.43	0.85

that in some contexts simply means unnecessary complexity, excessive fine tuning of a model to the data, even if this model does not need to generalize to other data, and even if the learning process is completely unsupervised. In such contexts, overfitting is simply the violation of the principle of Occam's razor, and assuming all the data is perfectly correct, the culprit is the learning algorithm. However, in our work this is not what we mean by overfitting.

In our work, overfitting is always linked to a supervised learning process, and always linked to either noise or outliers in the data, regardless of noisy observations being considered outliers or not, and regardless of the outliers being gross errors or simply exceptional (but true) observations. Overfitting is not equivalent to lack of generalization ability. True, when there is overfitting, there is no generalization ability, but not being able to generalize may simply mean there is not enough data to build a model. Overfitting implies that something was learned that should not have been learned, because learning it has resulted in a biased model that fails to explain similar but unseen data.

Particularly on this paper, due to the characteristics of our data, we do not focus on noise, but only on gross errors. One can argue that the learning algorithms are not overfitting, but simply fitting the data they were given, and that the origin of the problem is, instead, in the process of building the reference data. Nevertheless, these errors are outliers and they should not be learned. Therefore, we can state that the learning algorithms are indeed overfitting, and we make no distinction of whether the problem derives from the building of the data or from the learning process.

8.2. On the detection of overfitting

The main lesson to retain from the results section is that overfitting is not such a straightforward phenomenon to detect, and the present work illustrates this very well. Let us summarize the process adopted in this work: one reference data set is provided; this set is randomly split into training and test sets, several times, precisely to ensure that 1) the possible occurrence of overfitting does not go undetected and 2) the results are consistent across many independent executions of the learning method on different partitions of the data. Consistently low RMSE and high accuracy values indicate good learning, and the similarity of values measured on the training and test sets indicate good generalization. Still, overfitting was present, hidden, and it was only discovered during the validation of the models on a new data set.

There was nothing wrong with the process described in the previous paragraph. The training and test sets were drawn from the same distribution, as they should be, because they came from the same reference set. The problem is, same reference set, same errors. The models that perform well on the training data are the ones that learned the erroneous pixels, so they will perform equally well on the equally erroneous test data. The fact that these models (over)fit the errors is only detected on a new data set from a different source. This was the case in the present work, but there are two other possible scenarios.

The first alternative scenario is the test data coming from a different source and, unlike the training data, being free from labeling errors. In this case, the models that perform better on the (erroneous) training data will have a worse fitness on the (clean) test data, thus revealing overfitting. The second alternative scenario is tricky: the training data being free from errors, with only the test data containing erroneous data. Just like in the previous case, the models that perform better on the (clean) training data will have a worse fitness on the (erroneous) test data, thus revealing overfitting. Only this time there is no overfitting! At most, there may be some amount of true overfitting, but not in the amount suggested by the differences in the fitness measured on the training and test sets.

In summary, we have four different scenarios and possible outcomes: 1) clean training and test data: overfitting detected if and only if it is present; 2) erroneous training and test data (the present case): hidden overfitting not detected; 3) erroneous training data, clean test data: true overfitting correctly detected; 4) clean training data, erroneous test data: non-existent overfitting falsely detected.

It is not so obvious that case 3 above (erroneous training data, clean test data) would actually return overfitted models. We believe it would greatly depend on the amount and magnitude of the errors, as discussed in [3,104]. In the case of a very low percentage of errors, a robust method could be able to learn the general pattern of the training data without overfitting the errors (and therefore would obtain better fitness on the clean test set than on the erroneous training set[4]). However, this is not what happened in our problem. Despite a very low percentage of errors, overfitting did occur in both CART and StdGP, even if mostly hidden. And yet, using exactly the same faulty data sets, SSupGP was able to learn the general pattern as if the errors were not there. In the remainder of this section we explore the reasons why.

8.3. The effect of labeling errors in supervised and semi-supervised learning

First things first, we clear any doubts on whether the 20 erroneously labeled pixels, which represent less than 0.5% of the reference data, are indeed the culprit of the high misclassification errors shown in the results section. To this end, we correct the labels of these 20 pixels and perform a couple of sample runs with both CART and StdGP. Table 4 shows the results obtained by both methods on the two validation sets, with models learned with the error-free reference set. We immediately see the excellent accuracy and Kappa values that we were expecting in the first place. On the large validation set CART is able to outperform StdGP, and on the small validation set the results are the same for both methods.

In order to understand why the mislabeling errors undermine the success of supervised learning while the semi-supervised approach is apparently immune to them, we study the learning dynamics of both StdGP and SSupGP, performing separate measurements on the correct (correctly labeled) and erroneous (erroneously labeled) pixels, thus highlighting how differently the errors affect each of the learning methods.

Fig. 7 shows the evolution of the RMSE and overall accuracy during the StdGP evolution, for the correct and erroneous pixels separately (for this analysis we have put training and test data all together, since their RMSE are very similar - see Fig. 4 on Section 7). The figure shows boxplots for every 10th generation, from 0 to 200, calculated on the 30 runs. An extra line is also drawn to highlight the median. It is immediately obvious that the erroneous pixels are much more difficult to learn.

[4] In fact, many times we have observed such behavior in the past (in work not related to satellite imagery), but never thought of pointing the finger at the training data.

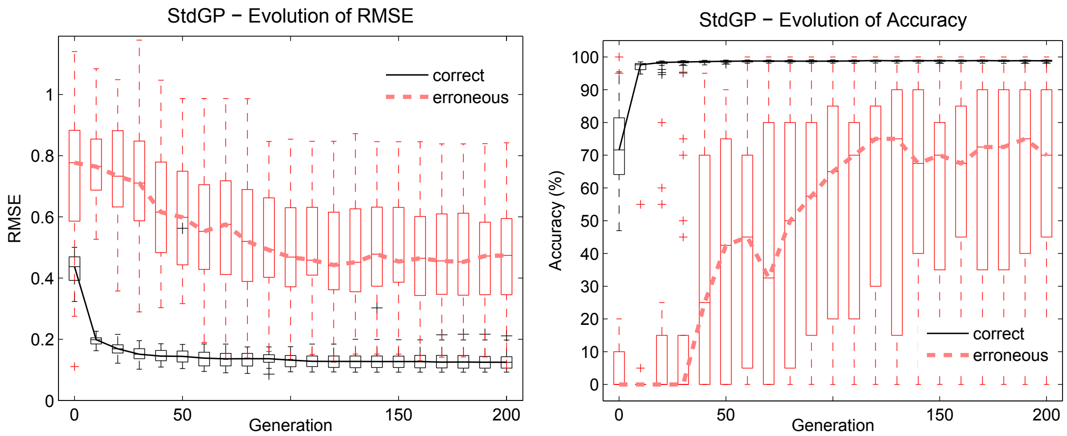

Fig. 7. Evolution of RMSE (left) and overall accuracy (right) during the StdGP evolution, for the correct and erroneous data separately. In the left plot there are four not shown "correct" outliers in generations 50–100 with approximate values 9, 16, 24 and 238, and 25 not shown "erroneous" outliers spread along all generations with values between 1 and 134.

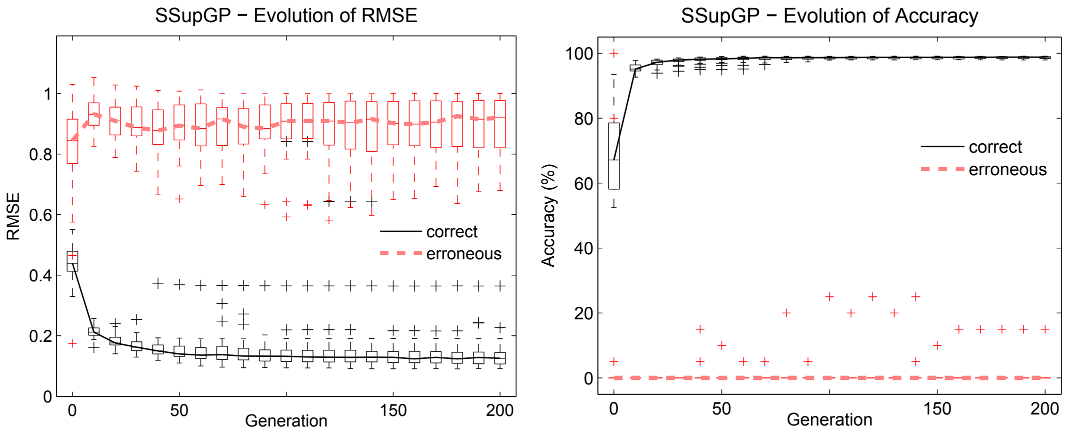

Fig. 8. Evolution of RMSE (left) and overall accuracy (right) during the SSupGP evolution, for the correct and erroneous data separately. In the left plot there are 17 not shown "correct" outliers in generations 50–200 with values between 1 and 479, and only one not shown "erroneous" outlier in generation 0 with approximate value 1.2.

Their median RMSE is much higher than the RMSE of the correct pixels, and even increases at some points of the evolution. Furthermore, there is a high variability of behavior among different runs, resulting in many outliers with high values that are not shown in the plot. In terms of the accuracy, the differences observed between correct and erroneous pixels are even more striking, with the correct pixels quickly reaching accuracy values close to 100% while the erroneous ones rise unevenly from 0% to around 75%.

Fig. 8 also shows the evolution of the RMSE and overall accuracy for the correct and erroneous pixels separately, but this time for SSupGP. The differences between correct and erroneous pixels are even larger than previously observed for StdGP. From the beginning of the run, the RMSE on the erroneous pixels simply does not decrease. On the contrary, as the RMSE on the correct pixels decreases, the RMSE on the erroneous ones tends to slightly increase. In terms of accuracy, once again the correct pixels quickly reach values close to 100%, but unlike StdGP, in SSupGP the erroneous pixels maintain a stubborn absolute 0% accuracy throughout the entire evolution (except for the shown outliers). It is clear that SSupGP does not learn the errors. Why?

8.4. Output values and distance to class labels

As mentioned earlier (Section 6), the main motivation for the SSupGP method was to avoid "wild" models whose output values fall very far from the interval [0 1], rewarding the well-behaved models that return values close to either 0 or 1. The secondary effect of this method turned out to be the apparent (and desirable) inability to learn the erroneous data. But what about the primary, the intended effect, of avoiding wild output values?

The purpose of the next two figures, in particular the comparison between them, is to observe whether SSupGP actually behaves substantially better than StdGP in terms of the "wildness" of its raw output values. Fig. 9 shows the dispersion of output values returned by all the 30 StdGP models for all the 30 sets of training (left) and test (right) data. All the models are relatively well behaved in the training data, with all output values inside the interval [-0.5 1.5] except for 18 (loosely called) outliers that fall in the interval [1.5 2]. Since this counting includes the output values of all 30 models, this represents less than one outlier per model. On the test data the dispersion of values was slightly higher, with 8 values falling in the interval [1.5 2], 7 other values falling in the interval [2127] (not shown in the plot), 4 values in the interval [−1 −0.5] and 4 other values in the interval [−2 −1] (not shown in the plot), therefore a total of 24 outliers, which also represents less than one outlier per model. Fig. 10 shows the dispersion of output values returned by all the 30 SSupGP models for all the 30 sets of training (left) and test (right) data. Curiously enough, SSupGP did not exactly show a more constrained behavior than StdGP. In the training data there were 10 values falling in the interval [1.5 2] and 3 others in the interval [2 2.5] (not shown in the plot), totaling 13 outliers (5 less than StdGP). In the test data there were 15 values in [1.5 2], 3 values in [2 25] (not shown in the plot) and a wild value of 32920 (not shown in the plot), plus 3 values in [−1 −0.5] and 2 values in [−7 −1] (not shown

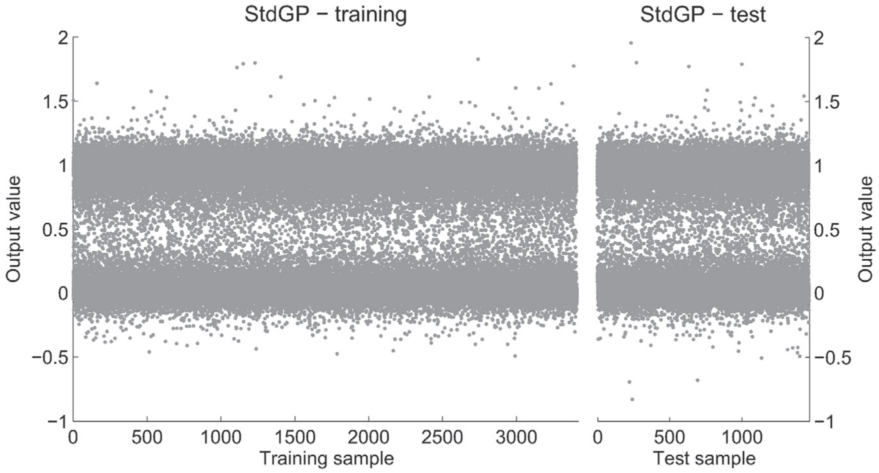

Fig. 9. Dispersion of output values returned by all the 30 StdGP models for all the 30 sets of training (left) and test (right) data. Some points are not shown in the test data, with values 127, 21.68, 11.67, 9.09, 5.29, 2.08, 2.04, −1.08, −1.23, −1.65 and −1.82.

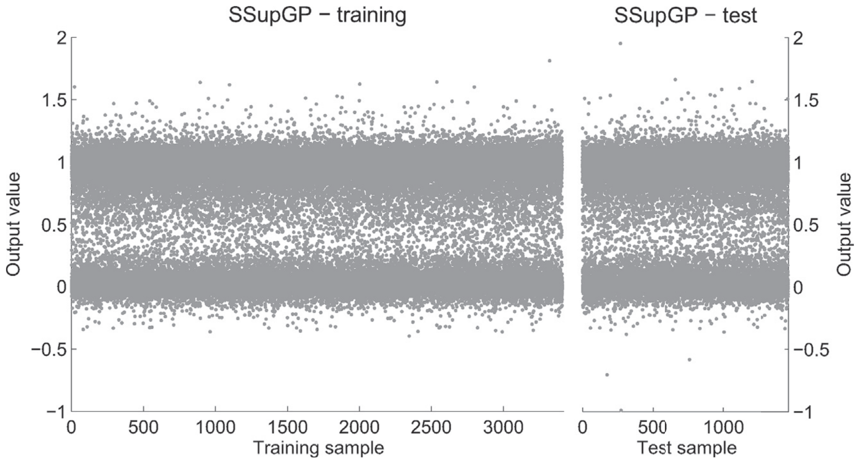

Fig. 10. Dispersion of output values returned by all the 30 SSupGP models for all the 30 sets of training (left) and test (right) data. Some points are not shown in the training data, with values 2.48, 2.40 and 2.24, and in the test data with values 32920, 24.23, 11.48, 2.39, −4.04 and −6.43.

in the plot), totaling 24 outliers (the same as StdGP). Apart from these slight differences, for SSupGP the range of values around 0 appears to be narrower than for StdGP. Although these are very small differences in behavior, they seem to be enough to produce a large difference in the range of the output values produced when classifying an entire image. However, it is still not clear how this difference affects the correctness of the classifications.

Fig. 11 shows the relationship between the distance to the (closest) class label and the accuracy of the classifications, presented as boxplots built with the values returned by the 30 models of StdGP (left) and SSupGP (right) on the reference set (training and test). The *x* axes are labeled in an unconventional manner: instead of completely specifying each interval below each box, they show only the interval endpoints between the boxes. The distances between 0 and 0.5 are grouped into 5 consecutive slots, all the same length. The distances larger than 0.5 are grouped differently, as they contain much fewer points. A distance larger than 0.5 means that either the output value is outside the interval [−0.5 1.5], or it is inside the interval and the classification is wrong. We have observed that for both StdGP and SSupGP there are no distances in the interval [23], and therefore we have grouped distances between 0.5 and 2 in one slot, and distances higher than 3 in another slot. The boxplots show that, for both StdGP and SSupGP, for distances until 0.5 the accuracy drops as the distance increases, from a median of almost 100% until as low as 60–65%. This was the expected behavior, already observed in similar studies [99]. However, for larger distances the behavior is different. For distances in the interval [0.5 2] the accuracy is again close or equal to 100%. This means that most of these classifications are correct, despite the output values of the models falling outside the [-0.5 1.5] interval. For distances higher than 3, the median accuracy is still maximum in StdGP, but only 25% in SSupGP. However, the medians in this last slot are calculated with very few points and therefore should not be used to derive any conclusions. What we can say is that StdGP and SSupGP once again exhibit very similar behavior in terms of output values produced by the models, and that the correctness of the classifications does not seem to be negatively affected by the presence of wild output values.

Performing a deeper exploration of the closeness of the output values to class labels, Fig. 12 shows boxplots of the maximum and minimum output values obtained during the evolution of the 30 models of StdGP (left) and SSupGP (right) for every 10th generation between 0 and 200. Extra lines highlight the evolution of the median values. Correct and erroneous pixels are treated separately. For the correct pixels, the behavior of StdGP and SSupGP is similar. In the first 10 generations the initial output values drift farther apart from each other, and then quickly stabilize for the rest of the evolution, around values that are slightly higher than 1 (for the maximum) and slightly lower than 0 (for the minimum). For the erroneous pixels the behavior is different, as expected. As StdGP learns part of them, the maximum output values

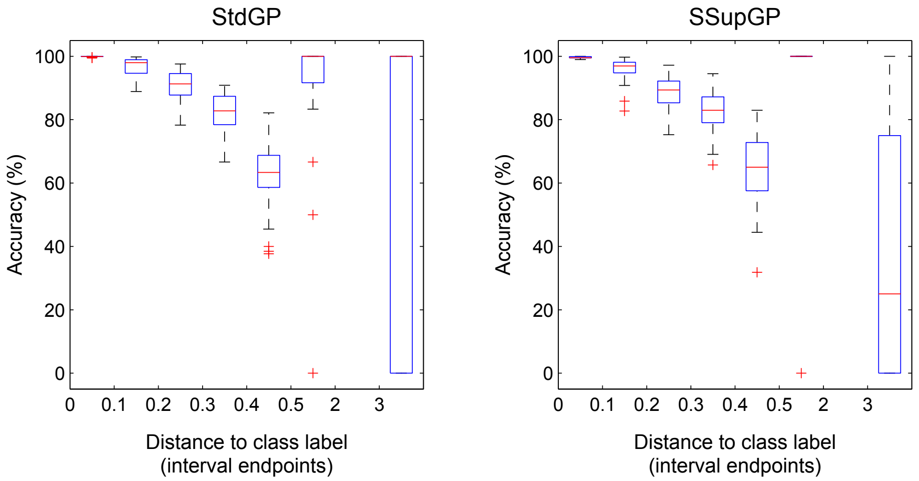

Fig. 11. Distance of the output values to the (closest) class label *versus* the accuracy of the classifications. Boxplots obtained with the values returned by the 30 models of StdGP (left) and SSupGP (right) on all the data (training and test) of the reference set. The unconventional *x* axes specify only the interval endpoints.

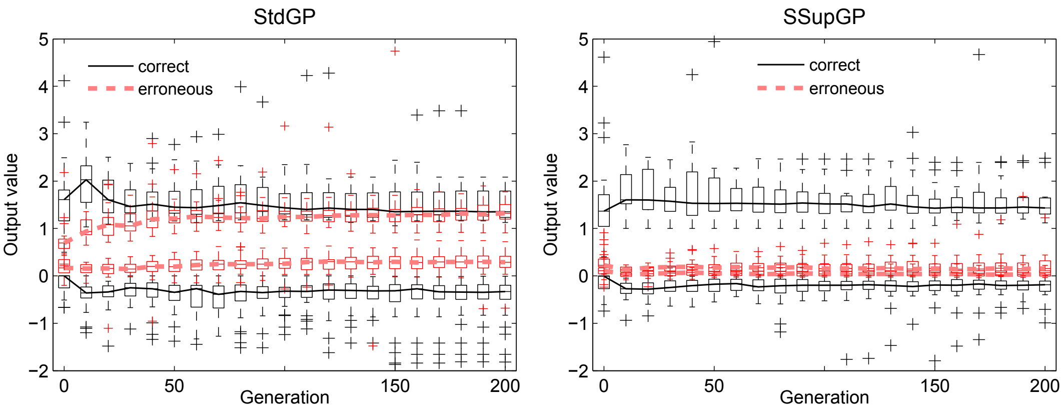

Fig. 12. Evolution of minimum and maximum output values obtained by the StdGP (left) and SSupGP (right) models, on the correct and erroneous data separately. For both StdGP and SSupGP there are additional outliers not shown in the plot, with values as low as −36 (StdGP) and −19 (SSupGP), and as high as 16586 (StdGP) and 33318 (SSupGP).

approximate the ones of the correct pixels, while the minimum output values remain slightly higher than 0 (therefore higher than the minimum output values of the correct pixels). On the contrary, with SSupGP the maximum and minimum values of the erroneous pixels remain very close to 0 during the entire evolution (and also higher than the minimum values of the correct pixels), as these pixels actually belong to class 0. Besides this obvious difference between StdGP and SSupGP, SSupGP tends to concentrate its values in narrower ranges and produce a lower number of outliers than StdGP. This was expected, and Figs. 9 and 10 had already partially revealed it. However, it is still a modest result that does not explain why SSupGP is so much better than StdGP.

8.5. The fitness function

At this point, we must assume that the general behavior of the models, in terms of output values and their distance to the class labels, is not related to the ability or inability to learn the erroneous data. We now look closely at the implications of the fitness function introduced by SSupGP, the modified RMSE (described in Section 6), with the goal of explaining why this fitness function does not lead the evolution into learning the erroneous data.

The first idea that comes to mind is that, by adding data to the calculation of the RMSE, the errors in the reference set are being diluted. This is only partially true. As the additional pixels do not contain any labels, they are not doing anything to contradict the erroneous information provided by the reference set. As far as labeled samples go, both StdGP and SSupGP are given the exact same percentage of errors. However, adding the unlabeled pixels to the calculation of the RMSE introduces a new way to improve fitness, which is minimizing the distance to any class label. As specified before (Section 7) the additional data represents around 70% of the amount of observations in the reference set. Considering that in each run 30% of the observations are taken for the test set, this means that the amount of training data is roughly the same as the amount of additional unlabeled data. Therefore, minimizing the distance to any class label becomes as important, and as influential to fitness, as minimizing the distance to a specific class label. It is well known that GP chooses the easiest way to improve fitness. In this case, we believe that the easiest way is to reduce the RMSE on the correct (and unlabeled) pixels, instead of forcefully learning a few pixels that obviously contradict the evolved models. Fig. 13 is an histogram of the shortest distances (0–0.1) to the class labels, calculated on the reference (labeled) data only, using the output values of all the 30 models evolved by each method. The unconventional *x* axes specify only the interval endpoints (as in Fig. 11). It clearly shows that the shortest distances are more frequent in the SSupGP models. In fact, more than half (52%) of the values returned by SSupGP models are closer than 0.02 from a class label (correct or incorrect), against only 38% of the values returned by StdGP.

We seem to have found the explanation for the success of SSupGP. Yet, one question still remains. If SSupGP improves fitness by minimiz-

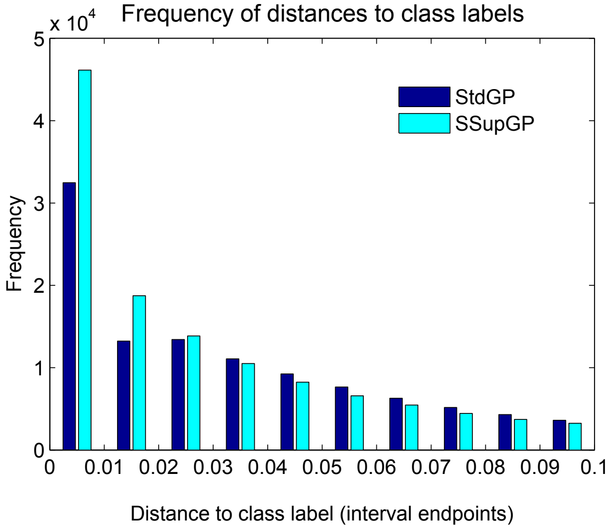

Fig. 13. Histogram of the shortest distances (0–0.1) to the class labels, calculated on the reference data using the output values of all the 30 models of each method. The unconventional *x* axes specify only the interval endpoints.

ing the distances to class labels, why is it that StdGP does not do the same? The answer is, it does. Looking at Fig. 7 (right), one can see that for the first 30 generations the median accuracy on the erroneous data remains 0%. But StdGP has less data to work with, less possibilities of easily improving fitness without making the effort of fitting the errors, so eventually it starts learning them. A striking observation is that already in generation 0 the random population of initial models finds it very difficult to fit the erroneous pixels, while easily fitting most of the correct ones. This observation is valid for both StdGP and SSupGP, and may have deep implications in future work (see Section 9).

9. Conclusions and future work

The story of this paper was very simple. Unusually bad results were obtained by two different methods, CART and GP, on a simple classification problem. A deeper look into the reference data used for learning revealed a small percentage of labeling errors, exposing and explaining a problem of hidden overfitting. At the same time, a modified fitness function was used in the GP method in order to promote a more constrained behavior of the evolved models, in terms of their range of raw output values. This new approach used extra unlabeled data, together with the original reference data, in a semi-supervised manner. The modified fitness function measures the error in the same way for both labeled and unlabeled data, but for the unlabeled points the class label is assumed to be the one closest to the raw output value. The models evolved with semi-supervised learning exhibited a very similar behavior to standard GP, and yet they did not learn the errors, completely avoiding overfitting.

This is the end of the story, but not the end of the work. This study has raised more questions than the ones it answered. The success of this semi-supervised GP approach is reasonably explained, but would it still hold if this classification problem was not so easy to solve? Or if the labeling errors affected more than just a tiny percentage of the reference data? In the absence of any errors, will this approach harm the learning process? This is a very important point, as we may not know beforehand if the reference data contains errors or not. We need to know if we should always use the semi-supervised method, in case of doubt.

If we do not possess extra data to implement this method, can we use synthetic data built on the same characteristics as the reference data? Can we even use the exact same reference data, as extra unlabeled data? This would be a kind of regularization. How much unlabeled data do we need in order to make the method work? And how much labeled data? Can a method be instructed to request more (labeled or unlabeled) data during the learning process, in case the models are not considered robust enough? This would enter the field of active learning. Can a method be instructed to request confirmation of the labels, in doubtful cases? Should the method simply discard the doubtful cases, assuming the labels are unreliable? Or should the method assume the existence of (and try to make the most of) fuzzy labels? Can we develop new fuzzy-supervised learning methods, where each label of the reference data has a confidence level attached? This confidence level could be either given by the human supervisor (in cases where there is no certainty, but only an educated guess) or resulting from a reasonable doubt regarding the given label, due to difficulties in learning (like in the present work). Would a well-thought method like this work better or worse than our very simple method, that simply relies on accepting what the evolving models see as more plausible?

Adding to the list of questions, can we expect this approach to be successful also in multiclass classification problems? Also when coupled with non-standard GP systems? Can it be adapted to regression problems? And finally, can we use the knowledge gained from this work in improving other machine learning methods?

Acknowledgements

The authors deeply thank the reviewers for their healthy skepticism and constructive comments, which really helped to improve the quality of this article. This work was partially funded by project PERSEIDS (PTDC/EMS-SIS/0642/2014) and BioISI RD unit (UID/MULTI/04046/2013) from the FCT/MCTES/PIDDAC, Portugal, and from the European Union’s Horizon 2020 Research and innovation programme under the Marie Skłodowska - Curie grant agreement No 691053.

References

[1] M.J. Carlotto, Effect of errors in ground truth on classification accuracy, Int. J. Remote Sens. 30 (18) (2009) 4831–4849.

[2] G.M. Foody, Assessing the accuracy of land cover change with imperfect ground reference data, Remote Sens. Environ. 114 (10) (2010) 2271–2285.

[3] G.M. Foody, The effect of mis-labeled training data on the accuracy of supervised image classification by SVM, in: 2015 IEEE International Geoscience and Remote Sensing Symposium (IGARSS), 2015, pp. 4987–4990.

[4] G.M. Foody, M. Pal, D. Rocchini, C.X. Garzon-Lopez, L. Bastin, The sensitivity of mapping methods to reference data quality: training supervised image classifications with imperfect reference data, ISPRS Int. J. Geo-Inf. 5 (11) (2016).

[5] J.R. Koza, Genetic Programming - on the Programming of Computers by Means of Natural Selection, Complex Adaptive Systems, MIT Press, 1993.

[6] R. Poli, W.B. Langdon, N.F. McPhee, A Field Guide to Genetic Programming, Lulu Enterprises, UK Ltd, 2008.

[7] P.B. Brazdil, K. Konolige, Learning from imperfect data, Mach. Learn. Meta-reason. Logics 1 (1) (1990) 207–232.

[8] V. Barnett, T. Lewis, Outliers in Statistical Data, second ed., John Wiley & Sons Ltd, 1978.

[9] B. Iglewicz, D. Hoaglin, How to Detect and Handle Outliers, ASQC Basic References in Quality Control, ASQC Quality Press, 1993. https://books.google.pt/books?id=siInAQAAIAAJ.

[10] X. Wu, X. Zhu, Mining with noise knowledge: error-aware data mining, IEEE Trans. Syst. Man Cybern. Part A 38 (4) (2008) 917–932.

[11] Z. Abedjan, X. Chu, D. Deng, R.C. Fernandez, I.F. Ilyas, M. Ouzzani, P. Papotti, M. Stonebraker, N. Tang, Detecting data errors: where are we and what needs to be done? Proc. VLDB Endow. 9 (12) (2016) 993–1004.

[12] J.I. Maletic, A. Marcus, Data Cleansing: a Prelude to Knowledge Discovery, Springer US, Boston, MA, 2010, pp. 19–32.

[13] X. Wang, M. Feng, Y. Wang, X.L. Dong, A. Meliou, Error diagnosis and data profiling with data x-ray, Proc. VLDB Endow. 8 (12) (2015) 1984–1987.

[14] K. Muşlu, Y. Brun, A. Meliou, Preventing data errors with continuous testing, in: Proceedings of the 2015 International Symposium on Software Testing and Analysis, ISSTA 2015, ACM, New York, NY, USA, 2015, pp. 373–384.

[15] H. Wang, M. Li, Y. Bu, J. Li, H. Gao, J. Zhang, Cleanix: a parallel big data cleaning system, SIGMOD Rec. 44 (4) (2016) 35–40.

[16] V. Damini, S. Rabindranath, K. Priyashree, K. Jayashubhaj, Optimized error detection analytics with bigdata on cloud, Int. J. Innov. Res. Sci. Eng. Technol. 5 (10) (2016) 238–242.

[17] B. Frénay, M. Verleysen, Classification in the presence of label noise: a survey, IEEE Trans. Neural Netw. Learn. Syst. 25 (5) (2014) 845–869, https://doi.org/10.1109/TNNLS.2013.2292894.

[18] V. Chandrashekar, S. Kumar, C.V. Jawahar, Image Annotation in Presence of Noisy Labels, Springer Berlin Heidelberg, Berlin, Heidelberg, 2013, pp. 381–389.

[19] H. Zhang, Z.-J. Zha, S. Yan, M. Wang, T.-S. Chua, Robust non-negative graph embedding: towards noisy data, unreliable graphs, and noisy labels, in: CVPR, IEEE Computer Society, 2012, pp. 2464–2471.

[20] V. Mnih, G.E. Hinton, Learning to label aerial images from noisy data, in: Proceedings of the 29th International Conference on Machine Learning, ICML 2012, Edinburgh, Scotland, UK, 2012.

[21] T. Yang, M. Mahdavi, R. Jin, L. Zhang, Y. Zhou, Multiple kernel learning from noisy labels by stochastic programming, in: Proceedings of the 29th International Conference on Machine Learning, ICML 2012, Edinburgh, Scotland, UK, 2012.

[22] L. Barbosa, J. Feng, Robust sentiment detection on twitter from biased and noisy data, in: Proceedings of the 23rd International Conference on Computational Linguistics: Posters, Association for Computational Linguistics, Stroudsburg, PA, USA, 2010, pp. 36–44.

[23] N. Natarajan, I.S. Dhillon, P.K. Ravikumar, A. Tewari, Learning with noisy labels, in: C.J.C. Burges, L. Bottou, M. Welling, Z. Ghahramani, K.Q. Weinberger (Eds.), Advances in Neural Information Processing Systems 26, Curran Associates, Inc, 2013, pp. 1196–1204.

[24] J. Bootkrajang, A. Kabán, Label-noise Robust Logistic Regression and its Applications, Springer Berlin Heidelberg, Berlin, Heidelberg, 2012, pp. 143–158.

[25] J. Bootkrajang, A. Kabán, Learning a Label-noise Robust Logistic Regression: Analysis and Experiments, Springer Berlin Heidelberg, Berlin, Heidelberg, 2013, pp. 569–576.

[26] C. Scott, S. Pozzi, G. Handy, M. Flaska, G. Blanchard, Classification with asymmetric label noise: consistency and maximal denoising, Electron. J. Stat. 10 (2) (2016) 2780–2824.

[27] T. Liu, D. Tao, Classification with noisy labels by importance reweighting, IEEE Trans. Pattern Anal. Mach. Intell. 38 (3) (2016) 447–461.

[28] G. Patrini, F. Nielsen, R. Nock, M. Carioni, Loss factorization, weakly supervised learning and label noise robustness, in: Proceedings of the 33rd International Conference on Machine Learning, 48, 2016, pp. 708–717. ICML'16, JMLR.org.

[29] Z. Lu, Z. Fu, T. Xiang, P. Han, L. Wang, X. Gao, Learning from weak and noisy labels for semantic segmentation, IEEE Trans. Pattern Anal. Mach. Intell. 39 (3) (2017) 486–500.

[30] S. Bhadra, M. Hein, Correction of noisy labels via mutual consistency check, Neurocomput 160 (C) (2015) 34–52.

[31] Y. Song, C. Wang, M. Zhang, H. Sun, Q. Yang, Spectral label refinement for noisy and missing text labels, in: Proceedings of the Twenty-ninth AAAI Conference on Artificial Intelligence, AAAI'15, AAAI Press, 2015, pp. 2972–2978.

[32] T.F.Y. Vicente, M. Hoai, D. Samaras, Noisy label recovery for shadow detection in unfamiliar domains, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 3783–3792.

[33] P.G. Ipeirotis, F. Provost, V.S. Sheng, J. Wang, Repeated labeling using multiple noisy labelers, Data Min. Knowl. Discov. 28 (2) (2014) 402–441.

[34] V. Franc, M. Sofka, K. Bartos, Learning Detector of Malicious Network Traffic from Weak Labels, Springer International Publishing, Cham, 2015, pp. 85–99.

[35] J. Efremova, A. Montes García, T. Calders, Classification of Historical Notary Acts with Noisy Labels, Springer International Publishing, Cham, 2015, pp. 49–54.

[36] P.J. Cano, E. Ramasso, Weighted Maximum Likelihood for Parameters Learning Based on Noisy Labels in Discrete Hidden Markov Models, Springer International Publishing, Cham, 2015, pp. 451–460.

[37] S.E. Reed, H. Lee, D. Anguelov, C. Szegedy, D. Erhan, A. Rabinovich, Training Deep Neural Networks on Noisy Labels with Bootstrapping, ICLR, Workshop abs/1412.6596, 2015.

[38] S. Sukhbaatar, R. Fergus, Training Convolutional Networks with Noisy Labels, ICLR, Workshop abs/1406.2080, 2015.

[39] S. Azadi, J. Feng, S. Jegelka, T. Darrell, Auxiliary Image Regularization for Deep Cnns with Noisy Labels, ICLR, Workshop abs/1511.07069, 2016.

[40] T. Xiao, T. Xia, Y. Yang, C. Huang, X. Wang, Learning from massive noisy labeled data for image classification, in: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 2691–2699, https://doi.org/10.1109/CVPR.2015.7298885.

[41] S. Yan, K. Chaudhuri, T. Javidi, Active learning from noisy and abstention feedback, in: 2015 53rd Annual Allerton Conference on Communication, Control, and Computing (Allerton), 2015, pp. 1352–1357.

[42] D. Henter, A. Stahl, M. Ebbecke, M. Gillmann, Classifier self-assessment: active learning and active noise correction for document classification, in: 2015 13th International Conference on Document Analysis and Recognition (ICDAR), 2015, pp. 276–280.

[43] G. Liu, Y. Yan, R. Subramanian, J. Song, G. Lu, N. Sebe, Active domain adaptation with noisy labels for multimedia analysis, World Wide Web 19 (2) (2016) 199–215.

[44] M.-R. Bouguelia, S. Nowaczyk, K.C. Santosh, A. Verikas, Agreeing to disagree: active learning with noisy labels without crowdsourcing, in: International Journal of Machine Learning and Cybernetics, 2017, pp. 1–13.

[45] F.E.B. Otero, M.M.S. Silva, A.A. Freitas, J.C. Nievola, Genetic Programming for Attribute Construction in Data Mining, Springer Berlin Heidelberg, 2003, pp. 384–393.

[46] J.L. Montaña, C.L. Alonso, C.E. Borges, J. de la Dehesa, Penalty Functions for Genetic Programming Algorithms, Springer Berlin Heidelberg, 2011, pp. 550–562.

[47] E.J. Vladislavleva, G.F. Smits, D. Den Hertog, Order of nonlinearity as a complexity measure for models generated by symbolic regression via pareto genetic programming, Trans. Evol. Comp. 13 (2) (2009) 333–349.

[48] L. Vanneschi, M. Castelli, S. Silva, Measuring bloat, overfitting and functional complexity in genetic programming, in: Proceedings of the 12th Annual Conference on Genetic and Evolutionary Computation, GECCO '10, ACM, New York, NY, USA, 2010, pp. 877–884.

[49] J. Fitzgerald, C. Ryan, On size, complexity and generalisation error in GP, in: Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation, GECCO '14, ACM, New York, NY, USA, 2014, pp. 903–910.

[50] O. Chapelle, B. Schlkopf, A. Zien, Semi-Supervised Learning, first ed., The MIT Press, 2010.

[51] Knowledge Extraction Based on Evolutionary Learning (KEEL), Semi-supervised Classification Data Sets Repository, 2017. http://sci2s.ugr.es/keel/semisupervised.php.

[52] Tyler (Tian) Lu, Fundamental Limitations of Semi-Supervised Learning(Master's thesis) 2009http://hdl.handle.net/10012/4387.

[53] I. Triguero, S. García, F. Herrera, Self-labeled techniques for semi-supervised learning: taxonomy, software and empirical study, Knowl. Inf. Syst. 42 (2) (2015) 245–284.

[54] A. Demiriz, K. Bennett, M.J. Embrechts, Semi-supervised clustering using genetic algorithms, in: In Artificial Neural Networks in Engineering (ANNIE-99), ASME Press, 1999, pp. 809–814.

[55] A. Demiriz, K.P. Bennett, M.J. Embrechts, A genetic algorithm approach for semi-supervised clustering, Int. J. Smart Eng. Syst. Des. 4 (1) (2002) 21–30.

[56] A. Klose, R. Kruse, Semi-supervised learning in knowledge discovery, Fuzzy Sets Syst. 149 (1) (2005) 209–233.

[57] Q. Zhang, S. Sun, Evolutionary classifier ensembles for semi-supervised learning, in: The 2010 International Joint Conference on Neural Networks (IJCNN), 2010, pp. 1–6.

[58] M.M. Adankon, M. Cheriet, Genetic algorithm–based training for semi-supervised SVM, Neural Comput. Appl. 19 (8) (2010) 1197–1206.

[59] X. Sun, D. Gong, W. Zhang, Interactive genetic algorithms with large population and semi-supervised learning, Appl. Soft Comput. 12 (9) (2012) 3004–3013.

[60] G. Lazarova, I. Koychev, A semi-supervised multi-view genetic algorithm, in: 2014 2nd International Conference on Artificial Intelligence, Modelling and Simulation, 2014, pp. 87–91.

[61] L. Dee Miller, L.-K. Soh, S. Scott, Genetic algorithm classifier system for semi-supervised learning, Comput. Intell. 31 (2) (2015) 201–232.

[62] S. Parsazad, E. Saboori, A. Allahyar, Data Selection for Semi-supervised Learning, CoRR abs/1208.1315. http://arxiv.org/abs/1208.1315.

[63] X. Zhang, L. Jiao, A. Paul, Y. Yuan, Z. Wei, Q. Song, Semisupervised particle swarm optimization for classification, Math. Prob. Eng. (2014):832135 11 pages.

[64] J. de Freitas, G.L. Pappa, A.S. da Silva, M.A. Goncalves, E. Moura, A. Veloso, A.H.F. Laender, M.G. de Carvalho, Active learning genetic programming for record deduplication, in: IEEE Congress on Evolutionary Computation, 2010, pp. 1–8.

[65] F. d. L. Arcanjo, G.L. Pappa, P.V. Bicalho, W. Meira Jr., A.S. da Silva, Semi-supervised genetic programming for classification, in: Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation, GECCO '11, ACM, New York, NY, USA, 2011, pp. 1259–1266.

[66] J.M. Fitzgerald, R.M.A. Azad, C. Ryan, Geml: evolutionary unsupervised and semi-supervised learning of multi-class classification with grammatical evolution, in: 2015 7th International Joint Conference on Computational Intelligence (IJCCI), 1, 2015, pp. 83–94.

[67] D. Robilliard, C. Fonlupt, Backwarding: an Overfitting Control for Genetic Programming in a Remote Sensing Application, Springer Berlin Heidelberg, 2002, pp. 245–254.

[68] R.G. Negri, S.J.S. Sant'Anna, L.V. Dutra, Semi-supervised remote sensing image classification methods assessment, in: 2011 IEEE International Geoscience and Remote Sensing Symposium, 2011, pp. 2939–2942, https://doi.org/10.1109/IGARSS.2011.6049831.

[69] C. Persello, L. Bruzzone, Active and semisupervised learning for the classification of remote sensing images, IEEE Trans. Geosc. Remote Sens. 52 (11) (2014) 6937–6956, https://doi.org/10.1109/TGRS.2014.2305805.

[70] U. Maulik, D. Chakraborty, Learning with transductive SVM for semisupervised pixel classification of remote sensing imagery, ISPRS J. Photogram. Remote Sens. 77 (2013) 66–78.

[71] Y. Liu, B. Zhang, L. min Wang, N. Wang, A self-trained semisupervised SVM approach to the remote sensing land cover classification, Comput. Geosci. 59 (2013) 98–107.

[72] L. Wang, S. Hao, Q. Wang, Y. Wang, Semi-supervised classification for hyperspectral imagery based on spatial-spectral label propagation, ISPRS J. Photogram. Remote Sens. 97 (2014) 123–137.

[73] K. Tan, E. Li, Q. Du, P. Du, An efficient semi-supervised classification approach for hyperspectral imagery, ISPRS J. Photogram. Remote Sens. 97 (2014) 36–45.

[74] M. Dalponte, L.T. Ene, M. Marconcini, T. Gobakken, E. Næsset, Semi-supervised SVM for individual tree crown species classification, ISPRS J. Photogram. Remote Sens. 110 (2015) 77–87.

[75] K. Tan, J. Hu, J. Li, P. Du, A novel semi-supervised hyperspectral image classification approach based on spatial neighborhood information and classifier combination, ISPRS J. Photogram. Remote Sens. 105 (2015) 19–29.

[76] M. Romaszewski, P. Głomb, M. Cholewa, Semi-supervised hyperspectral classification from a small number of training samples using a co-training approach, ISPRS J. Photogram. Remote Sens. 121 (2016) 60–76.

[77] X. Ma, H. Wang, J. Wang, Semisupervised classification for hyperspectral image based on multi-decision labeling and deep feature learning, ISPRS J. Photogram. Remote Sens. 120 (2016) 99–107.

[78] A.E. Melchiori, A.W. Setzer, F. Morelli, R. Libonati, P. d. A. Cândido, S. C. d. Jesús, A Landsat-TM/OLI Algorithm for Burned Areas in the Brazilian Cerrado: Preliminary Results, Imprensa da Universidade de Coimbra, Coimbra, 2014, pp. 1302–1311.

[79] A. Bastarrika, E. Chuvieco, M.P. Martín, Mapping burned areas from landsat tm/etm+ data with a two-phase algorithm: balancing omission and commission errors, Remote Sens. Environ. 115 (4) (2011) 1003–1012.

[80] N. Koutsias, M. Pleniou, G. Mallinis, F. Nioti, N.I. Sifakis, A rule-based semi-automatic method to map burned areas: exploring the usgs historical landsat archives to reconstruct recent fire history, Int. J. Remote Sens. 34 (20) (2013) 7049–7068.

[81] Y. Liu, Q. Dai, J. Liu, S. Liu, J. Yang, Study of burn scar extraction automatically based on level set method using remote sensing data, PLoS One 9 (2) (2014) 1–11.

[82] J.M. Pereira, A.C. Sá, A.M. Sousa, J.M. Silva, T.N. Santos, J.M. Carreiras, Spectral characterisation and discrimination of burnt areas, in: Remote Sensing of Large Wildfires, Springer Berlin Heidelberg, 1999, pp. 123–138.

[83] L. Breiman, Classification and Regression Trees, Chapman & Hall/CRC, 1984.

[84] C. Brodley, M. Friedl, Decision tree classification of land cover from remotely sensed data, Remote Sens. Environ. 61 (3) (1997) 399–409.

[85] M. Pal, P. Mather, An assessment of the effectiveness of decision tree methods for land cover classification, Remote Sens. Environ. 86 (4) (2003) 554–565, https://doi.org/10.1016/S0034-4257(03)00132-9.

[86] S. Kotsiantis, Decision trees: a recent overview, Artif. Intell. Rev. 39 (4) (2013) 261–283, https://doi.org/10.1007/s10462-011-9272-4.

[87] M.-S. Park, M. Kim, M.-I. Lee, J. Im, S. Park, Detection of tropical cyclone genesis via quantitative satellite ocean surface wind pattern and intensity analyses using decision trees, Remote Sens. Environ. 183 (2016) 205–214, https://doi.org/10.1016/j.rse.2016.06.006.

[88] J.M.C. Pereira, M.J.P. Vasconcelos, A.M. Sousa, A Rule-based System for Burned Area Mapping in Temperate and Tropical Regions Using NOAA/AVHRR Imagery, Springer Netherlands, Dordrecht, 2000, pp. 215–232.

[89] J. Silva, J. Pereira, A. Cabral, A. Sá, M. Vasconcelos, B. Mota, J.-M. Grégoire, An estimate of the area burned in southern africa during the 2000 dry season using spot-vegetation satellite data, J. Geophys. Res. Atmos. 108 (D13) (2003) 8498.

[90] J. Silva, A. Sá, J. Pereira, Comparison of burned area estimates derived from spot-vegetation and landsat etm+ data in africa: influence of spatial pattern and vegetation type, Remote Sens. Environ. 96 (2) (2005) 188–201.

[91] J.R. Koza, Human-competitive results produced by genetic programming, Genet. Program. Evol. Mach. 11 (3–4) (2010) 251–284.

[92] A. Makkeasorn, N.-B. Chang, J. Li, Seasonal change detection of riparian zones with remote sensing images and genetic programming in a semi-arid watershed, J. Environ. Manag. 90 (2) (2009) 1069–1080.

[93] J. dos Santos, C. Ferreira, R. da S. Torres, M. Gonçalves, R. Lamparelli, A relevance feedback method based on genetic programming for classification of remote sensing images, Inf. Sci. 181 (13) (2011) 2671–2684.

[94] C. Puente, G. Olague, S.V. Smith, S.H. Bullock, A. Hinojosa-Corona, M.A. González-Botello, A genetic programming approach to estimate vegetation cover in the context of soil erosion assessment, Photogram. Eng. Remote Sens. 77 (4) (2011) 363–376.

[95] M. Trabucchi, C. Puente, F.A. Comin, G. Olague, S.V. Smith, Mapping erosion risk at the basin scale in a mediterranean environment with opencast coal mines to target restoration actions, Reg. Environ. Change 12 (4) (2012) 675–687.

[96] N.-B. Chang, Z. Xuan, Y.J. Yang, Exploring spatiotemporal patterns of phosphorus concentrations in a coastal bay with modis images and machine learning models, Remote Sens. Environ. 134 (2013) 100–110.

[97] J. Almeida, J.A. dos Santos, W.O. Miranda, B. Alberton, L.P.C. Morellato, R. da S. Torres, Deriving vegetation indices for phenology analysis using genetic programming, Ecol. Inf. 26 (Part 3) (2015) 61–69.

[98] D.J. Lary, A.H. Alavi, A.H. Gandomi, A.L. Walker, Machine learning in geosciences and remote sensing, Geosci. Front. 7 (1) (2016) 3–10 special Issue: Progress of Machine Learning in Geosciences.

[99] S. Silva, M.J. Vasconcelos, J.B. Melo, Bloat free genetic programming versus classification trees for identification of burned areas in satellite imagery, in: European Conference on the Applications of Evolutionary Computation, Springer Berlin Heidelberg, 2010, pp. 272–281.

[100] S. Luke, L. Panait, Lexicographic parsimony pressure, in: Proceedings of the Genetic and Evolutionary Computation Conference, GECCO '02, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2002, pp. 829–836.

[101] S. Silva, S. Dignum, L. Vanneschi, Operator equalisation for bloat free genetic programming and a survey of bloat control methods, Genet. Program. Evol. Mach. 13 (2) (2012) 197–238.

[102] G.M. Foody, Status of land cover classification accuracy assessment, Remote Sens. Environ. 80 (1) (2002) 185–201.

[103] J.A. Richards, X. Jia, Remote Sensing Digital Image Analysis: an Introduction, third ed., Springer-Verlag New York, Inc, Secaucus, NJ, USA, 1999.

[104] G.M. Foody, Ground reference data error and the mis-estimation of the area of land cover change as a function of its abundance, Remote Sens. Lett. 4 (8) (2013) 783–792.