A Many-Objective Optimization Approach for Complexity-based Data set Generation

Thiago R. França*, Péricles B.C. Miranda*, Ricardo B.C. Prudêncio†, Ana C. Lorena‡, André C.A. Nascimento*
*Departamento de Computação, Universidade Federal Rural de Pernambuco
†Centro de Informática, Universidade Federal de Pernambuco
‡Divisão de Ciência da Computação, Instituto Tecnológico de Aeronáutica

***Abstract*—The assessment of machine learning algorithms in a particular task is usually done by means of empirical evaluation on real world observational data. However, sometimes there is no previously annotated data available. Synthetic datasets have gained attention as an alternative for efficient classifier evaluation, since they are accessible and high parameterizable for a given learning task. The characterization of such databases can be done by means of descriptors on the learning object at hand, e.g., complexity measures, which extract statistical and geometric characteristics from the data sets, for a given classification problem, in order to estimate their complexity. Such complexity measures can be used to guide the production of synthetic datasets, so it reinforces one or more dataset features. The present work proposes the use of a many-objective algorithm for the generation of synthetic data considering four measures of complexity that will be balanced at the same time. The results showed that the proposal is able to optimize conflicting objectives, generating datasets of specific complexities.**

***Index Terms*—Dataset generation, Many-objective optimization, Complexity measures**

I. Introduction

Currently, several classification algorithms are being used in a wide variety of tasks, distributed in many areas, such as medicine, security, business, and education. Due to the large heterogeneity of such tasks and the particularities that make a particular algorithm suit better than others, it is necessary to analyze the quality of the classifier in terms of classification performance (accuracy, precision, recall, and others), computational cost interpretability [1]. These empirical analyses support the specialist in the selection of the most appropriate classifier for its particular problem.

A common practice to assess the performance of one or more classifiers is to train and test them in different databases, most of them publicly available. It has been observed that such practice may have its limitations [2]. The first is that the classifier's performance estimate may be associated with its own limitation or with the complexity of the data set itself (for example, unrepresentative or missing data, high-class imbalance, and others). Testing different algorithms on the same data sets may seem to overcome this problem, as the complexity of the data set is shared by all algorithms considered. However, this can also lead to misleading conclusions, as there are complexities in the data set that may affect the learning algorithms in different ways. The second limitation is that, even if the obtained results are consistent when compared with hundreds of real-world data sets, the results still remain specific to the data sets considered in the experimental evaluation. Trying to extend the information obtained from these analyses to a different data set can be ineffective. In addition, obtaining more real databases, with different characteristics, to evaluate classifiers in different aspects, is a complicated and often costly task.

In view of the aspects presented above, synthetic databases have gained attention as an alternative for efficient classifier evaluation, since they are accessible and parameterizable [1]. Although synthetic databases are not directly associated with specific real data sets, they can be used as representations of large classes of data [2]. Due to the parameterizable nature of the synthetic bases, it is possible to create a database in which its features follow, for example, a given Pearson correlation, making it possible to analyze the behavior of classifiers in this scenario.

Different strategies have been proposed for the systematic generation of synthetic databases for classification problems. A common approach is to sample examples following a specific distribution [2]. Let us suppose that we want to create a synthetic database from a normal distribution, with different mean and variance for the different classes. By modifying the values of the means and variances, it is possible to build different databases, with distinct rates of overlapping classes.

Another approach proposes to generate synthetic databases by modifying the geometric structure of the data [3], which can be done using complexity measures [3]–[6]. These complexity measures were introduced by Ho and Basu [7], and have been adopted in different types of analysis in recent years [8]–[11]. Complexity measures extract statistical and geometric characteristics from the data sets, for a given classification problem, in order to estimate their complexity. In such scenarios, some studies have proposed the optimization of different measures in order to generate synthetic databases with different levels of complexity. This is usually achieved by the application of optimization algorithms (single and multi-objective) for the generation of the synthetic data sets. There are approaches that create a new synthetic database [3], [4] or that make a sampling from pre-existing data sets [5], [6], to reach a given value of one or more measures of complexity.

This work proposes an optimization method for the generation of new synthetic databases taking into account four different measures of complexity: Error rate of linear classifier (L2), the fraction of borderline points (N1), the ratio of

978-1-7281-6929-3/20/$31.00 ©2020 IEEE

average intra/inter nearest neighbor (NN) distance (N2) and the class imbalance ratio (C2). It is worth noting that the related works, so far, have considered at most three measures of complexity to be optimized in the generation process. As major contributions of this work, we point out: i) use of a many-objective optimization algorithm (MaOA) to generate synthetic data; ii) proposition of an experimental setup for the use of MaOA for the complexity-based data generation problem.

The proposed method was evaluated on the generation of synthetic datasets composed of 100 instances with 2 and 10 attributes. Results demonstrate that the problem is challenging, and the proposal is able to optimize conflicting objectives, generating datasets of specific complexities.

This article is structured as follows: Section II introduces complexity measures and many-objective optimization, important concepts in this work. Section III presents some related work which consider complexity measures in the generation of synthetic data. Section IV details the proposed approach. Section V presents the experimental methodology used to evaluate the proposed method. Section VI presents the experimental results, and Section VII highlights the conclusions and future work.

II. Background

This section introduces fundamental concepts about data complexity and many-objective optimization, aiming for a better understanding of the proposed approach.

A. Data Complexity

It is a known fact in the machine learning community that the performance of a classifier is strongly dependent on the nature of the data used in the training and evaluation analysis [1]. Given that, the analysis of data complexity is a task that has achieved great relevance in understanding the complexity-performance relationship, how to overcome this dependency, and how to improve the classifier's performance.

The work developed by [7] is considered one of the pioneers in proposing complexity measures (descriptors) to characterize a given classification problem, in order to estimate its difficulty. These measures have been used lately for different purposes, such as characterizing the domain of competence of different machine learning algorithms [9]; description of classification problems in meta-learning studies [11]; and generation of new classification databases [12].

According to the work of [1], the main complexity measures can be grouped into the following categories:

- *Feature-based measures*: quantify how informative for the class separation the available features are;
- *Linearity measures*: quantify whether the problem classes are linearly separable;
- *Neighborhood measures*: quantify the presence and density of identical and different classes in local neighborhoods, which can be useful in capturing the shape of the decision boundaries and characterizing class overlap;
- *Network measures*: extract structural dataset information by modeling it as a graph;
- *Dimensionality measures*: quantify the sparcity of the data based on the number of samples related to the dimensionality of the data;
- *Class imbalance measures*: quantifies the ratio between the number of examples in each class.

Complexity measures are estimated from a given data set D that contains n pairs of examples (x_i, y_i), where $x_i = (x_{i1}, ..., x_{im})$ and $y_i \in 1, ..., n_c$ are the instances and their respective labels. Each instance x_i is defined by m attributes, and has a y_i label associated to one of the n_c classes.

B. Many-Objective Optimization

Multi-objective optimization problems having more than three conflicting objectives are also known as many-objective (MaO) problems. The MaO tries to find solutions that satisfy all the conflicting objectives ($\vec{f}$) at the same time. MaO can be characterized as the problem of; given a vector of functions ($\vec{f}$) whose components represent objective functions to be optimized, the goal is to find a vector of decision variables that fulfills the defined constraints and also optimizes $\vec{f}$ at the same time.

More specifically, consider the search space of solutions defined as $\mathcal{S}$, and a set of objective functions $\vec{f}(\vec{x}) := [f_1(\vec{x}), f_2(\vec{x}), ..., f_k(\vec{x})]$, where $\vec{x} = (x_1, x_2, ..., x_n) \in \mathbb{R}^n$ is the vector of parameters in $\mathcal{S}$. The optimization task is to find a set of solutions $S^{opt} \subset \mathcal{S}$, based on the vector of objective functions $\vec{f}(\vec{x})$. A general MaO minimization problem can be defined as:

$$\text{minimize } \vec{f}(\vec{x}) := [f_1(\vec{x}), f_2(\vec{x}), ..., f_k(\vec{x})], \tag{1}$$

subject to:

$$g_l(\vec{x}) \leq 0 \quad l = 1, 2, ..., p, \tag{2}$$

$$h_j(\vec{x}) = 0 \quad j = 1, 2, ..., q, \tag{3}$$

where $\vec{x} = (x_1, x_2, ..., x_n) \in \mathbb{R}^n$ is a vector in the decision search space; k is the number of objectives and $g_l(\vec{x})$ and, $h_j(\vec{x})$ are the objective functions and $p + q$ is the number of objectives of the problem.

In a traditional multi-objective optimization problem (MOP), Pareto dominance is usually used as a base of comparison between solutions. Algorithms that use such a strategy have demonstrated promising results when applied to MOPs with up to three objectives [13]. Nevertheless, the ability to find optimal solutions of such methods typically worsens when applied in a MaOPs scenario, i.e., in a setting more than three objectives [13], [14]. This can be explained by the increase of the dimensionality of the search space, as the probability of a solution to be dominated by other solutions in the population is reduced [13]. This leads to a dramatical reduction in performance, making these algorithms to behave as bad as, or even worse, than random search approaches [13], [15]

Another aspect intrinsically related to MaOPs is that the number of non-dominated solutions to approximate the Pareto front rises exponentially as the number of objective functions increases. Thus, it is required to have a bigger population to store the nondominated solutions [16]. A more detailed list of these limitations can be found in [13], [17].

Due to the aspects cited above, several approaches have been proposed in the Evolutionary Multi-Objective Optimization (EMO) to overcome the limitations of MOP-based methods when applied to MaOPs [13]. Among them, we highlight the Non-dominated Sorting Genetic Algorithm, third version (NSGA-III). The NSGA-III algorithm was proposed by Jan and Deb [18], and it consists of an extended version of the widely used NSGA-II (for MOPs) to handle MaOPs. The NSGA-III uses a reference point approach, with a non-dominated sorting mechanism. Besides, it uses a clustering operator to promote the diversity of the solutions when leading with MaOPs. This algorithm has demonstrated promising results when applied in MaOPs.

III. Complexity-based Dataset Generation

The task of synthetic data set generation based on complexity measures consists in labeling instances of the classification problem, trying to satisfy one or more complexity measures. Figure 1 presents an overview of the process. Initially, a database without labels (generated or pre-existing) is provided for the optimization algorithm. The optimization algorithm aims to find the best combination of label instances (solution) present in the search space, which satisfies the specified complexity measures. This task is a traditional combinatorial optimization problem [6]. Since the number of possible combinations grows exponentially as the number of instances increases, using an exhaustive method, in situations where the database has a large number of instances, becomes impractical [6]. Therefore, different studies used heuristics and/or metaheuristics to generate synthetic data.

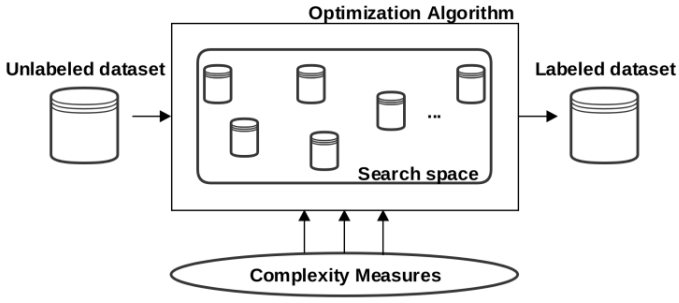


Fig. 1. Complexity-based dataset generation pipeline.

The first work that considered complexity measures in the generation of synthetic databases was developed by [8]. In this seminal work, a search heuristic that best met the fraction of borderline points (N1) complexity measure was used. In [4], a genetic algorithm (GA) was used, in which each individual of the population consisted of a vector of classes/labels, with each class associated with an example from the database. In this work, GA was used to find the combination of labels that best satisfied the complexity measure fraction of borderline points (N1). As an extension of the previous work, Màcia et al. [3] proposed the use of a multi-objective genetic algorithm (MOGA) for the labeling of examples in a classification database. In this work, three measures of complexity were considered in the labeling process, i.e., the fraction of borderline points (N1), the ratio of average intra/inter nearest neighbor distance (N2), and the ratio of the maximum Fisher's discriminant (F1). It is worth to note that this approach used the same representation of individuals and genetic operators that [4].

The authors in [5] proposed the generation of synthetic databases through MOGA, as well as in [3], but taking into account other complexity measures. In this case, the following were considered: volume of the overlapping region (F2), the nonlinearity of nearest neighbor (NN) classifier (N4), and the fraction of maximum covering spheres (T1). In addition, in order to avoid the generation of the entire database, the authors propose to perform the instance selection based on known base classification problems. The idea is to select the subset of examples that optimize a combination of the values of the complexity measures. In this case, MOGA has some restrictions to control the number of selected examples, class proportions, and data duplicity. Although the data sets produced to cover the space of complexity in a more realistic way, the algorithm can be considered expensive and has many parameters to be adjusted.

A recent work, proposed by [6], used the mono-objective Hill-climbing greedy search algorithm for the generation of synthetic data. However, the generation process takes into account an initial database, where the labels of the pairs of examples are exchanged iteratively. With this, you can change the structure of the database, but preserving some of its initial characteristics, such as the number of examples, the classes, and the data distribution within the classes.

The generation of synthetic databases is relevant and still has several aspects that need to be further investigated. One of them is the analysis of different complexity measures in the optimization process. Another important aspect is to consider the usage of optimization algorithms with more than three objective functions. In this case, the problem is considered many-objective and would need specific optimization algorithms (for example, the Non-dominated Sort Genetic Algorithm-III) to produce the desired databases.

The present work proposes the use of a many-objective algorithm for the generation of synthetic data considering four measures of complexity that will be balanced at the same time. Table I summarizes each related work, comparing them with the one proposed in this paper. The aim is to make our contributions clear in relation to the previous works.

IV. Many-Objective Synthetic Data set Generator

This paper proposes the generation of synthetic data sets by adopting a MaOA to optimize different complexity measures.

TABLE I
CONTRASTING RELATED WORKS WITH THE PROPOSAL.

Reference	Approach	#Objectives	Objective Function
Màcia et al. (2008) [8]	Not informed search heuristic	1	Fraction of borderline points (N1)
Màcia et al. (2008) [4]	GA	1	Fraction of borderline points (N1)
Màcia et al. (2009) [3]	NSGA II	3	Fraction of borderline points (N1), Ratio of average intra/inter nearest neighbor distance (N2), Ratio of the maximum Fisher's discriminant (F1)
Màcia et al. (2010) [5]	NSGA II	3	Volume of overlapping region (F2), Non linearity of NN classifier (N4), Fraction of maximum covering spheres (T1)
de Melo et al. (2018) [6]	Hill-climbing greedy	1	F1, N1 and Error rate of NN classifier (N3) at a time
Proposal	NSGA III	4	Error rate of linear classifier (L2) Fraction of borderline points (N1) the ratio of average intra/inter NN distance (N2) Imbalance ratio (C2)

In the following sections, each step of the generation process (see the pipeline in Figure 1) is presented and discussed.

A. Instances Generation

The first step in the proposal is to initialize a data set from scratch, given the number of predictor attributes, instances, and classes. In this step, the attributes' values are randomly sampled from a uniform distribution in the $[0, 1]$ interval. The class attribute's values are defined in the next steps according to the optimization process.

Details about the number of attributes, instances, and classes adopted in the experiments are provided in Section V.

B. Individual Representation

Let D be a data set with n instances. Each instance x_i in D has a label $y_i \in 1, ..., n_c$, in which n_c is the number of classes. In our work the class labels are assigned by the MaOA. In this way, an individual $\vec{I}$ is represented by a vector of length n, in which each position assumes a class value y_i:

$$\vec{I} = (y_1, y_2, ..., y_n). \tag{4}$$

Hence, each individual represents a possible combination of class labels for the data set. If we have D with $n = 100$ instances and $n_c = 3$ possible class labels, the resulting search space has 3^{100} possible solutions.

C. Individual Evaluation

As discussed in Section II-B, in a scenario of many-objectives, each individual $\vec{I}$ is associated with a vector of fitness values derived from the objective functions in $\vec{f}$. In the context of synthetic data generation, $\vec{f}$ is the chosen set of complexity measures. The choice of the objective functions considered in the optimization process depends on the specialist's demand in terms of complexity. In this work, specifically, four measures of complexity were chosen for the definition of objective functions: a measure of linearity, two neighborhood-based measures, and a measure of class balance. Each measure is described as follows:

Error rate of linear classifier (L2): Measure L2 calculates the error rate of the linear SVM classifier. This measure is calculated using the following equation:

$$L2 = \frac{\sum_n^{i=1} I(h(x_i) \neq y_i)}{n}. \tag{5}$$

Fraction of borderline points (N1): this measure estimates the size and complexity of the necessary decision boundary identifyed the critical points in the database - those very close to each other, but belonging to different classes. This measure is calculated using the following equation:

$$N1 = \frac{1}{n} \sum_{i=1}^{n} I((x_i, x_j) \epsilon MST \wedge y_i \neq y_j). \tag{6}$$

where MST is the Prim's minimum spanning tree algorithm.

Ratio of average intra/inter NN distance (N2): this measure calculates the ratio between two sums, i) the sum of the distances between each example and its closest neighbor in the same class (intra-class), ii) the sum of the distances between each example and its closest neighbor in another class (inter-class). This measure is calculated using the following equation:

$$intra\_extra = \frac{\sum_{i=1}^{n} d(x_i, NN(x_i) \epsilon y_i)}{\sum_{i=1}^{n} d(x_i, NN(x_i) \epsilon y_j \neq y_i)}. \tag{7}$$

Imbalance ratio (C2): is an index that measures the classes imbalance in the data set. Here, the definition proposed by [19] was adopted, as it can also be applied to multi-class problems. This index is calculated using the following equation:

$$C2 = 1 - \frac{1}{IR}, \tag{8}$$

where,

$$IR = \frac{n_c - 1}{n_c} \sum_{i=1}^{n_c} \frac{n_{c_i}}{n - n_{c_i}}. \tag{9}$$

Each of the complexity measures produces a value in the $[0, 1]$ range. The simpler the problem, the closer to 0 it is, and closer to 1 otherwise. For example, if the measure $L2 = 0$, it means that the data is completely linearly separable, that is, a

simple classification problem. On the other hand, if $L2 = 1$, it means that the data is not linearly separable to the extreme.

To calculate the fitness value of a candidate solution (label combination), each of the complexity measures is calculated, taking into account the labeled solution database. The objective functions defined here are the distance between the value obtained by each complexity measure, subtracted by the value of complexity desired by the specialist. Thus, there are four objective functions (f_{L2}, f_{N1}, f_{N2}, and f_{C2}) to minimize the distance between the value calculated by the measure and the expected value. The four objective functions are defined in the following equations:

$$f_{L2} = |target\_L2 - L2|, \tag{10}$$

$$f_{N1} = |target\_N1 - N1|, \tag{11}$$

$$f_{N2} = |target\_N2 - N2| \, and, \tag{12}$$

$$f_{C2} = |target\_C2 - C2|, \tag{13}$$

where, $target\_L2$, $target\_N1$, $target\_N2$ and $target\_C2$ are target values defined by a specialist.

D. Optimization Process

The MaOA used in this work was the NSGA-III, since it is suitable for the context of many objective functions (more than three). As can be seen in the Algorithm 1, NSGA-III follows the traditional procedure of evolutionary algorithms. The first step is the creation of an initial population, followed by an iterative process of selecting the most promising individuals, and by the application of genetic operators, such as crossover and mutation. Then, the new individuals are evaluated, and the Pareto front is updated using the fast_nondominated_sorting procedure. The difference between NSGA-III and NSGA-II is in the last step. It was necessary to replace the NSGA-II crowding distance operator with a clustering operator. The clustering operator associates members of the population with fixed cluster centroid provided by a well-distributed set of landmarks (Z^r). This clustering operator was introduced to promote the diversity of solutions in MaOPs. Finally, the output of the NSGA-III is a Pareto composed of non-dominated solutions, that is, incomparable in relation to the four objectives considered here. Details on NSGA-III operators and parameters used in this work will be presented in Section V-A.

V. Experimental Environment

In this section, we present the experimental setup and the methodology used to assess the proposed method. The simulations were executed on a laptop with an Intel Core i7-8750H with 4M cache memory, 2.2 GHz clock speed, and 8GB RAM. The programming languages used to implement the proposed approach were Python™and R; the following libraries were used: Scikit learn [20] for the machine learning use, DEAP [21] for the use of the many-objective optimization algorithm and ECoL [1] which contains implemented complexity measures to characterize classification problems. These libraries are reliable and extensively used for research projects.

Algorithm 1: MOO algorithm.

PARAMETERS:
Z^r: reference points
initialize_population()

while *!stop_criterion* **do**
  selection_technique()
  genetic_operations()
  objectives_evaluation()
  fast_nondominated_sorting()
  clustering_operation()
end

TABLE II
Parameters used to create and optimize the data sets.

Parameter	Value
Sample size (n)	100
Number of features (m)	2; 10
Number of classes (n_c)	2
Number of runs	10
Complexity measures	L2; N1; N2; C2
Initialization method	Uniform random
Population size	30
#Generations	5000
Selection method	Tournament
Reproduction rate	0.6
Mutation rate	0.2
Crossover rate	0.9
Crossover method	Two-point
Mutation method	Shuffle Indexes

A. Experimental Setup

The step that precedes the optimization process is the generation of the synthetic database. The synthetic base, size n, was created following a uniform distribution of values between 0 and 1, with a given number of attributes m. Table II presents the parameters used to generate the synthetic database. We tested two different sample sizes, numbers of features, and classes, considering four complexity measures.

Once the synthetic bases are created, the optimizer will try to find the combination of labels that best satisfy the different targets for each of the complexity measures. It is important to mention that each execution is independent, but uses the same synthetic data set.

In this work, we used the NSGA-III as the many-optimization algorithm, and it is available in the DEAP library [21]. All parameters were chosen empirically, and they are stated in Table II. The optimizer used 5,000 iterations as a stop criterion, generating up to $150,000$ synthetic data sets.

TABLE III
COMPLEXITY TARGET VALUES FOR EACH PROBLEM LEVEL.

Problem level	L2	N1	N2	C2
Easy	0.07	0.07	0.07	0.07
Medium	0.07	0.07	0.07	0.48
Difficult	0.22	0.22	0.22	0.22
Very difficult	0.35	0.35	0.35	0.48

B. Evaluation Methodology

To evaluate the proposed method, the experiment was divided into two parts: 1) data visualization and 2) quantitative analysis. The first considers a synthetic database with 2 attributes and 100 samples. The optimization algorithm was used to generate synthetic data sets in four different complexity scenarios: easy, medium, difficult, and very difficult. The target values for each problem level can be seen in Table III. The easy problem presents a linearly separable behavior, balanced and with a low level of overlapping. The medium problem is linearly separable and with low overlapping, but with highly unbalanced behavior. The difficult problem presents overlapping; the data are not linearly separable and present some imbalance. Finally, the very difficult problem is chaotic, with a high level of non-linearity, imbalance, and overlapping. The objective of this experiment is to show (graphically through scatter plots) the data set labeled with a promising solution obtained by the optimization algorithm at each of the problem levels. As the output of the NSGA-III is a Pareto composed by a set of incomparable solutions, we used the Borda count method [22] to select a single promising solution from the Pareto. This method is a single-winner ranking method in which every criterion ranks each algorithm, and then an average rank is returned, where the algorithm in the first place is the winner. The solution returned by the borda count is used for the data visualization.

The second experiment has a more quantitative bias, evaluating the performance of the optimizer in a data set with 100 samples considering 2 and 10 attributes. In this experiment, the fitness values, the execution time, and the number of iterations used by the optimizer to achieve the results are presented. Besides, we also executed classifiers to investigate their performance (in terms of average accuracy) in the different generated data sets.

VI. RESULTS AND DISCUSSION

A. Data Visualization

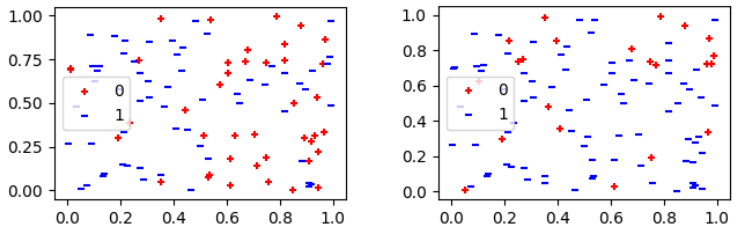
Figure 2 shows the position of each instance of the synthetic data set generated with their respective labels, optimized by NSGA-III. The x and y axes are the values of the data set attributes. Figures 2-A, -B, -C, and -D represent one of the solutions belonging to the resulting Pareto, found by NSGA-III, for each of the problem levels (see Table III). This experiment considers the generation of data sets with 100 instances and 2 attributes. Figure 2-a shows the data set with its labels optimized to meet the objectives of the simple problem. The simple problem is extreme, where all objectives need to be optimized for the smallest values of complexity. As the measures of complexity L2, and N1 and N2 are conflicting, minimizing them at the same time is a difficult task. The values achieved for each of the objectives were, $L2 = 0.21$, $N1 = 0.53$, $N2 = 0.35$ and $C2 = 0.06$. The algorithm achieved a value close to the optimum in terms of $C2$, turning the base balanced. Although the generated data set has a low value of $L2$, it does not have a completely linearly separable behavior. This is due to the balance of classes, making the task of reducing overlapping (values of $N1$ and $N2$) more difficult.

Figure 2-b shows the data set with its labels optimized to meet the objectives of the average problem (see Table III). In this case, an attempt is made to optimize the value of $C2 = 0.48$, increasing the base imbalance. The values achieved for each objective of the average problem were, $L2 = 0.15$, $N1 = 0.40$, $N2 = 0.17$ and $C2 = 0.48$. The optimization algorithm was able to optimize the value of $C2$ as much as possible, making the base unbalanced. The imbalance favors the reduction of overlapping since there are fewer instances of a given class. For this reason, the values achieved for each of the other objectives were lower when compared to the simple problem.

Figure 2-c shows the data set with its labels optimized to meet the objectives of the difficult problem. In this case, an attempt is made to increase the measurement values, increasing the complexity of the data set (see Table III). The values achieved by the algorithm for each objective of the difficult problem were, $L2 = 0.21$, $N1 = 0.54$, $N2 = 0.32$ and $C2 = 0.21$.

Figure 2-d shows the data set with its labels optimized to meet the objectives of the very difficult problem. Like the simple problem, the very difficult problem is extreme, only that all objectives need to be optimized to the highest values of complexity. As the complexity measures L2, and N1 and N2 are conflicting, optimizing their values at the same time is not trivial. In this problem, the goal is to find a solution that approximates $L2 = 0.35$, $N1 = 0.35$, $N2 = 0.35$ and $C2 = 0.48$. However, due to the conflicting nature of the objectives, the values reached for each one of them were, $L2 = 0.17$, $N1 = 0.60$, $N2 = 0.72$ and $C2 = 0.48$. The result achieved shows a data set with chaotic behavior, with a high rate of unbalance and overlapping.

The optimization algorithm was also used to generate data sets with 10 attributes. The objective is to check if there is any impact on the optimization of complexity measures when the number of attributes increases. Table IV shows the optimization results for both 2 and 10 attributes. As you can see, the increase in the number of attributes did not have a negative impact on the generation of data sets. The execution time is close, and in some situations, the values achieved are even better when compared to the results for those of 2 attributes. The values in bold mean that they are better than the other.



(a) L2:0.21,N1:0.53,N2:0.35,C2:0.06 (b) L2:0.15,N1:0.40,N2:0.17,C2:0.48

(c) L2:0.21,N1:0.54,N2:0.32,C2:0.21 (d) L2:0.17,N1:0.45,N2:0.35,C2:0.48

Fig. 2. L2, N1, N2 and C2 values of the generated data sets with 100 instances and 2 attributes.

TABLE IV
L2, N1, N2 AND C2 VALUES OF THE GENERATED DATA SETS WITH 100 INSTANCES CONSIDERING 2 ATTRIBUTES AND 10 ATTRIBUTES.

Data set complexity	**m**	**Target** ($L2$, $N1$, $N2$, $C2$)	**Reached** ($L2$, $N1$, $N2$, $C2$)	Time (s)
Simple	2	(0.07, 0.07, 0.07, 0.07)	(0.21, **0.53**, 0.35, 0.06)	2.460
	10	(0.07, 0.07, 0.07, 0.07)	(**0.16**, 0.58, **0.31**, **0.07**)	2.475
Medium	2	(0.07, 0.07, 0.07, 0.48)	(0.15, **0.40**, **0.17**, **0.48**)	1.470
	10	(0.07, 0.07, 0.07, 0.48)	(**0.14**, 0.45, 0.26, 0.50)	1.488
Difficult	2	(0.22, 0.22, 0.22, 0.22)	(**0.21**, **0.54**, 0.32, 0.21)	1.950
	10	(0.22, 0.22, 0.22, 0.22)	(0.20, 0.55, **0.30**, **0.22**)	2.010
Very Difficult	2	(0.35, 0.35, 0.35, 0.48)	(**0.17**, **0.45**, **0.35**, **0.48**)	2.190
	10	(0.35, 0.35, 0.35, 0.48)	(0.16, 0.46, 0.38, 0.47)	2.215

B. Quantitative Analysis

To complement the previous analysis, classifiers (from the Scikit Learn library) were run, with their default parameters, on the data sets generated with 100 attributes and 2 classes. Figure 3 shows the average accuracy achieved (x-axis) for each classifier at different levels of problems (y-axis). The accuracy of the classifiers was calculated through the cross-validation experiment ($cv = 10$), being performed 100 times to calculate the mean and standard deviation.

The behavior of the Decision Tree and Naive Bayes classifiers showed a pattern, even though they have different learning schemes. As the complexity of the problem increased, the accuracy achieved by these classifiers also decreased. This shows the influence that the chosen complexity measures have on the performance of these classifiers, and can help in estimating their accuracy.

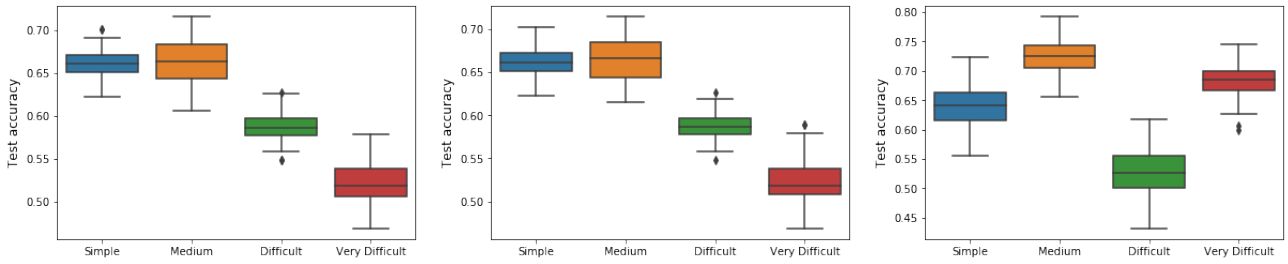
The Random Forest algorithm behaved differently from the other two. As it is an ensemble capable of dealing with non-linearity and data imbalance, its performance in medium and very difficult problems was superior to low or intermediate balancing problems.

The task of optimizing different and conflicting measures of complexity is not a non-trivial task. This work is the first to treat the data set generation problem as a many-objective problem. It is important to mention that the results showed here were not compared to the related work because it would be unfair to compare approaches which try to optimize a different number of objective functions.

The results presented in this section showed that although the task is difficult, the proposed method was able to generate data sets of specific complexities, trying to satisfy the different objectives as much as possible. The experiments carried out gave an idea of the potential of the proposal, but several analyses still need to be carried out, such as the generation of data sets with more instances, attributes, classes; optimization of other different measures of complexity; and variation of genetic operators optimization algorithm.

VII. CONCLUSION AND FUTURE WORKS

Machine learning algorithms are commonly assessed using empirical evaluation on real-world data. Though, sometimes there is no previously labeled data accessible. Synthetic data sets have gained recognition as an alternative for efficient classifier evaluation because they are available and high parameterizable for a given learning task. The characterization of such data sets can be done employing descriptors on the



(a) Decision Tree (b) Naive Bayes (c) Random Forest

Fig. 3. L2, N1, N2 and C2 values of the generated data sets with 100 instances and 2 attributes.

learning object at hand, e.g., complexity measures, which extract statistical and geometric characteristics from the data sets, for a given classification problem, to assess their complexity. Such complexity measures can be used to guide the creation of synthetic data sets. The present work proposes the use of a many-objective algorithm for the generation of synthetic data considering four measures of complexity that will be optimized at the same time. The main contributions of this work are: i) use of a many-objective optimization algorithm (MaOA) to generate synthetic data; ii) proposition of an experimental setup for the use of MaOA for the complexity-based data generation problem.

Results demonstrate that the problem is challenging, and the proposal is able to optimize conflicting objectives, generating data sets of specific complexities. The experiments carried out gave an idea of the potential of the proposal, but several analyses still need to be carried out. As future work, we intend to evaluate different many-objective optimization algorithms for the data set generation; optimize other different complexity measures and investigate their impact on the search for solutions; generate synthetic data sets with a greater number o instances, attributes and classes; and evaluate different distributions for the creation of unlabeled data. Finally, we also intend to build a meta-base to support the recommendation of suitable classifiers for classification problems with specific characteristics of complexity.

REFERENCES

[1] A. C. Lorena, L. P. Garcia, J. Lehmann, M. C. Souto, and T. K. Ho, "How complex is your classification problem? a survey on measuring classification complexity," *ACM Computing Surveys (CSUR)*, vol. 52, no. 5, pp. 1–34, 2019.

[2] D. R. Amancio, C. H. Comin, D. Casanova, G. Travieso, O. M. Bruno, F. A. Rodrigues, and L. da Fontoura Costa, "A systematic comparison of supervised classifiers," *PloS one*, vol. 9, no. 4, 2014.

[3] N. Macia, A. Orriols-Puig, and E. Bernado-Mansilla, "Beyond hand-made artificial data sets," in *International Conference on Hybrid Artificial Intelligence Systems*. Springer, 2009, pp. 605–612.

[4] ——, "Genetic-based synthetic data sets for the analysis of classifiers behavior," in *2008 Eighth International Conference on Hybrid Intelligent Systems*. IEEE, 2008, pp. 507–512.

[5] ——, "In search of targeted-complexity problems," in *Proceedings of the 12th annual conference on Genetic and evolutionary computation*, 2010, pp. 1055–1062.

[6] V. V. de Melo and A. C. Lorena, "Using complexity measures to evolve synthetic classification datasets," in *2018 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2018, pp. 1–8.

[7] T. K. Ho and M. Basu, "Complexity measures of supervised classification problems," *IEEE transactions on pattern analysis and machine intelligence*, vol. 24, no. 3, pp. 289–300, 2002.

[8] N. Macia, E. Bernado-Mansilla, and A. Orriols-Puig, "Preliminary approach on synthetic data sets generation based on class separability measure," in *2008 19th International Conference on Pattern Recognition*. IEEE, 2008, pp. 1–4.

[9] J. Luengo and F. Herrera, "An automatic extraction method of the domains of competence for learning classifiers using data complexity measures," *Knowledge and Information Systems*, vol. 42, no. 1, pp. 147–180, 2015.

[10] L. P. Garcia, A. C. de Carvalho, and A. C. Lorena, "Effect of label noise in the complexity of classification problems," *Neurocomputing*, vol. 160, pp. 108–119, 2015.

[11] ——, "Noise detection in the meta-learning level," *Neurocomputing*, vol. 176, pp. 14–25, 2016.

[12] N. Macia and E. Bernado-Mansilla, "Towards uci+: A mindful repository design," *Information Sciences*, vol. 261, pp. 237–262, 2014.

[13] H. Ishibuchi, N. Tsukamoto, and Y. Nojima, "Evolutionary many-objective optimization: A short review," in *Evolutionary Computation, 2008. CEC 2008.(IEEE World Congress on Computational Intelligence). IEEE Congress on*. IEEE, 2008, pp. 2419–2426.

[14] R. C. Purshouse and P. J. Fleming, "Evolutionary many-objective optimisation: An exploratory analysis," in *Evolutionary Computation, 2003. CEC'03. The 2003 Congress on*, vol. 3. IEEE, 2003, pp. 2066–2073.

[15] M. Garza-Fabre, G. Toscano-Pulido, and C. A. C. Coello, "Two novel approaches for many-objective optimization," in *Evolutionary Computation (CEC), 2010 IEEE Congress on*. IEEE, 2010, pp. 1–8.

[16] H. Ishibuchi, Y. Sakane, N. Tsukamoto, and Y. Nojima, "Evolutionary many-objective optimization by nsga-ii and moea/d with large populations," in *Systems, Man and Cybernetics, 2009. SMC 2009. IEEE International Conference on*. IEEE, 2009, pp. 1758–1763.

[17] B. Li, J. Li, K. Tang, and X. Yao, "Many-objective evolutionary algorithms: A survey," *ACM Computing Surveys (CSUR)*, vol. 48, no. 1, p. 13, 2015.

[18] K. Deb and H. Jain, "An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part i: Solving problems with box constraints." *IEEE Trans. Evolutionary Computation*, vol. 18, no. 4, pp. 577–601, 2014.

[19] A. K. Tanwani and M. Farooq, "Classification potential vs. classification accuracy: a comprehensive study of evolutionary algorithms with biomedical datasets," in *Learning Classifier Systems*. Springer, 2009, pp. 127–144.

[20] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg *et al.*, "Scikit-learn: Machine learning in python," *Journal of machine learning research*, vol. 12, no. Oct, pp. 2825–2830, 2011.

[21] F.-A. Fortin, F.-M. De Rainville, M.-A. Gardner, M. Parizeau, and C. Gagné, "DEAP: Evolutionary algorithms made easy," *Journal of Machine Learning Research*, vol. 13, pp. 2171–2175, jul 2012.

[22] D. Black, "Partial justification of the borda count," *Public Choice*, vol. 28, no. 1, pp. 1–15, 1976.