A Hybrid Surrogate Model for Evolutionary Undersampling in Imbalanced Classification

Hoang Lam Le, Dario Landa-Silva, Mikel Galar, Salvador Garcia, I. Triguero

***Abstract*—Data preprocessing is a key stage in data mining that allows machine learning algorithms to obtain meaningful insights. Many preprocessing problems such as feature selection or instance selection can be modelled as optimisation/search problems. Evolutionary algorithms have traditionally excelled in this task when dealing with data of a moderate size. However, their application to large datasets typically involves very high computational costs. In this work, we propose a hybrid surrogate model for evolutionary undersampling in imbalanced classification problems. These are characterised by having a highly skewed distribution of classes in which evolutionary algorithms aim to balance the training data by selecting only the most relevant data. The proposed technique combines a two-stage clustering-based surrogate method with a windowing approach to quickly approximate fitness values of the chromosomes and accelerate the search. The experiments carried out in 44 standard imbalanced datasets show that the proposed hybrid surrogate model highly reduces the computational cost of the evolutionary algorithm without a considerable loss of performance.**

***Index Terms*—Data Preprocessing, Evolutionary undersampling, Surrogate models, Imbalanced classification, Fitness approximation, Windowing**

I. Introduction

In data science, preprocessing techniques [1] aim to transform raw data into the so-called Smart Data [2], which is data in a usable shape to allow the subsequent machine learning to be successful. Among others, data preprocessing includes data cleaning, dimensionality reduction, instance reduction and discretisation. Many of these strategies have been formulated as optimisation problems, so that, a search algorithm finds a preprocessed dataset that enables machine learning to extract useful knowledge from the data [3]. Evolutionary algorithms have widely been used in data preprocessing problems such as feature selection [4] or instance selection [5] with very promising results.

In this work, we are interested in the class imbalance problem for classification, which is a recurrent issue in data science, in which the input data has a severely skewed distribution of classes [6]. Considering two-class datasets, the problem happens when the number of positive class examples (typically the class of interest) is very limited with respect to the negative ones. Under these circumstances, canonical classification techniques may be biased towards the majority class and may also have to deal with another series of difficulties such as overlapping, small sample size, or small disjuncts. Several approaches have been designed to tackle this problem, which can be divided into three main groups: data sampling, algorithmic modifications and cost-sensitive solutions [7]. These models have also been successfully combined with ensemble learning algorithms [8].

Evolutionary undersampling (EUS) [9] is a data sampling technique, based on instance selection, that performs a binary search to balance the distribution of classes of the original dataset by removing examples of the negative class. This search is carefully guided by the CHC algorithm [10] that aims to increase the performance on the two classes of the problem while reducing the number of negative examples. However, when dealing with large datasets, the search may become very time-consuming due to the cost associated to fitness evaluation (consisting of classifying the entire training set with the resulting preprocessed dataset). Recently, the processing time of EUS has been reduced using distributed approaches in big data platforms [11] that require larger computational infrastructures.

This work is focused on reducing the computational cost of EUS by means of fitness approximation approaches [12], such as surrogate models [13], [14]. These methods may reduce the computational cost of search algorithms by accelerating fitness evaluation of each chromosome, as opposed to parallelisation techniques that merely focus on reducing processing time. Whilst there are many surrogate models for continuous search problems, methods for combinatorial domains remain under-explored [15]. Two simple solutions based on partitioning the training set can be used to reduce the runtime of EUS: stratification [16] and windowing [17]. While the former helps reduce the processing time, the latter actively reduces the computational cost of the fitness evaluation considering subsets of training data for fitness evaluation. In [18], we proposed a clustering-based surrogate model for EUS, called EUSC. As opposed to windowing or stratification, EUSC considers the entire training data when computing fitness values. However, EUSC only performs real evaluations for a limited number of chromosomes.

In this paper, we propose a hybrid surrogate model, called a hybrid surrogate model for EUS (EUSHC), that integrates windowing with EUSC to highly reduce the computational cost of the fitness function without misleading the search

H. Lam Le, D. Landa-Silva and I. Triguero are with the Computational Optimisation and Learning (COL) Lab, School of Computer Science, University of Nottingham, United Kingdom. E-mails: {hoang.le,dario.landasilva,isaac.triguero}@nottingham.ac.uk

M. Galar is with the Institute of Smart Cities, Public University of Navarre, Campus Arrosadía s/n, 31006 Pamplona, Spain. E-mail: mikel.galar@unavarra.es

S. Garcia is with the Department of Computer Science and Artificial Intelligence of the University of Granada, Granada, Spain, 18071. E-mail: salvagl@decsai.ugr.es

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

for accurate solutions. First, a two-stage clustering process allows us to transform binary chromosomes into real coding chromosomes, so that, distances between solutions can be computed effectively [18]. Then, the proposed hybrid method uses windowing to estimate the fitness value of a reduced number of chromosomes, and the fitness values of the rest of solutions are approximated based on their similarity and imbalanced ratio. Thus, the entire search is guided by approximate fitness values. In our experiments with 44 standard imbalance datasets, we show how this greedy surrogate model allows for a massive reduction of computational costs without considerably reducing accuracy.

The paper is organised as follows: Section II introduces the background of the research topic, consisting of EUS and its specification. In Section III, we describe the proposed hybrid surrogate model for EUS. Section IV analyses the empirical results. Finally, Section V summarises the conclusions.

II. Background

In this section we briefly describe the topics covered in this paper. Section II-A presents related work on imbalanced classification. Then, Section II-B details the EUS algorithm. Finally, Section II-C discusses different techniques to accelerate evolutionary algorithms.

A. Imbalanced classification

In binary classification, a problem is classed as imbalanced when it is significantly composed of more instances from one class than from the other. Standard classification algorithms usually measure success based on accuracy rate (percentage of correctly classified examples), which neglects the prediction performance of the minority class. The most common alternatives in this scenario are the Area Under the ROC Curve (AUC) and the g-mean. The AUC (Area Under the ROC-Curve) measures how well the trade off between true positive ($TPrate$) and false positive rates (FP_{rate}). A popular approximation [6] of this measure is given by

$$AUC = \frac{1 + TP_{rate} - FP_{rate}}{2}. \quad (1)$$

The g-mean computes the geometric mean between the true positive rates and true negative rates (TN_{rate}):

$$GM = \sqrt{TP_{rate} \cdot TN_{rate}} \quad (2)$$

Both measures are usually considered interchangeably and extensively used in numerous experimental studies with imbalanced datasets.

Solutions tackling the class imbalanced problem can be grouped into data preprocessing [2], [19] and algorithmic modification [20]. Those operating at the algorithmic level update existing learning algorithm to acknowledge the imbalanced situation, while those at data-level preprocessing modify the data to make the class distribution less critically unequal. Cost-sensitive learning [21] is a kind of algorithm-level modification that aims to learn more characteristics of the minority class examples by adding a higher penalty for their misclassification. Ensemble-based methods are gaining momentum, combining an ensemble learning algorithm (e.g. Bagging, Boosting) [22] with data preprocessing or cost-sensitive techniques.

Data-level preprocessing strategies allow us to use any classifier after their application. They can be roughly split into undersampling or oversampling. Undersampling focuses on the majoirty class samples, eliminating redundant examples. Conversely, oversampling aims at creating arficial data for the minority class. Hybrid methods combine both approaches [23]. Although all of these approaches are proved effective in many studies, oversampling and hybrid methods tend to generate additional data, which result in a higher computational cost. Undersampling is particularly interesting when dealing with large and big dataset as it reduces the size of the data. Hence, the corresponding classifier can be applied faster.

EUS [9] is an interesting alternative for undersampling that carefully select the majority class samples (as opposed to Random undersampling), so that, we obtain a more balanced dataset that preserve or even improve the final performance. As such, EUS is an instance selection algorithm that acts only on the majority class samples. Despite its performance, it is well known that its practical application is typically limited to relatively small datasets. To address this situation, researchers have focused on designing big data solutions to enable EUS to tackle big datasets in a reasonable time [11], [24]. Whilst these solutions are needed to handle really big datasets, they do not reduce the computational cost of EUS, but they distribute the computation across a number of computers. In this work, we are interested in reducing the computational cost of EUS, which could later be integrated with distributed solutions to handle big datasets.

B. Evolutionary undersampling for imbalanced classification

This section presents the details of EUS algorithm. Let N^- and N^+ be the number of majority and minority class samples in a two-class dataset. Each instance x_i has m-dimensions and belongs to a class given by x_{i_ω}, $x_i = (x_{i_1}, x_{i_2}, x_{i_3}, ..., x_{i_m}, x_{i_\omega})$. EUS performs a search using an evolutionary algorithm, namely CHC [10], to optimise the resulting reduced set RS of training samples that are later used by a classifier. EUS encodes the selection of majority class samples with a binary representation $\{1,\ 0\}$. Thus, the size of a chromosome is N^- as only instances in the majority class are examined for elimination. All minority class samples are always part of RS. A chromosome is expressed as: $chr_j = (v_{x_1}, v_{x_2}, v_{x_3}, , v_{x_{N^-}})$ where $v_{x_i} \in \{1, 0\}$ indicates whether sample x_i is included or not.

EUS keeps a population of NP chromosomes that are assessed and ranked based on their quality. To do so, a fitness function is used to evaluate the quality of the chromosome based on how well the chromosome balances the class distributions and an expected performance of the selected instances. To do so, a RS is constructed based on the selection of majority class samples determined by the chromosome plus all minority class examples. Then, the entire training dataset is classified based on RS as training data. Similarly to most

previous works, in this paper we adopt the Nearest Neighbour (NN), k=1, [25] rule as base classifier. As performance measure, the g-mean is applied (defined in Eq. (2)).

The complete fitness function looks like this:

$$f_{chr_j} = \begin{cases} GM_{chr_j} - \left|1 - \frac{N^+}{s^-}\right| \cdot P & \text{if } s^- > 0 \\ GM_{chr_j} - P & \text{if } s^- = 0, \end{cases} \quad (3)$$

where s^- is the number of selected negative instances and P is a penalisation factor that focuses on the balance between both classes. P is typically set to 0.2 as recommended by the authors, since it provides a good trade-off between both objectives.

Thus, evaluating the quality of each chromosome may become a long operation when the size of the training set grows. Therefore, our research question is: *Can we develop a fast EUS that quickly approximates the fitness value of chromosomes without misleading the search?*

C. Surrogate models for fitness approximation

In the optimisation field, many studies have investigated the acceleration of fitness evaluations. We can find solutions such as delta evaluation (based on only computing changes in the solution and estimate cost) or fitness inheritance (which is inspired by the idea that an offspring can also inherit a fitness value from its parents, not only its own genes) [12].

Machine learning techniques such as clustering and supervised learning have also been used to approximate fitness values. Clustering algorithms aim to decrease the number of fitness evaluations by splitting the entire population (based on the chromosome representation) into a number of groups. Then, the chromosomes closest to the clusters' centres are evaluated by the exact function, while other cluster members are approximated according to their distance to the evaluated solutions [26]. Supervised learning techniques aim to create a predictive model that can approximate the fitness function. The model is adjusted based on the data points accumulated from the evaluation history. Such a data-driven model is built under the assumption that there is continuity among data points, at which a small variation in decision variables will cause a smooth change in the fitness value. Both solutions work well on continuous optimisation problems but their application to combinatorial search spaces is currently under-explored.

III. EUSHC: Hybrid Surrogate Model for Evolutionary Undersampling

This section presents the proposed hybrid method for EUS that combines windowing and a clustering-based surrogate model. Section III-A discusses the motivation behind this approach. Section III-B describes the windowing component and Section III-C details the hybrid approach.

A. Motivation

As stated before, EUS has demonstrated to be a very effective solution to determine the best subset of majority class elements that tackles the imbalanced situation in a classification problem. The cost associated to its fitness evaluate motivates the use of approximations to assess the quality of a chromosome. The main problem lies in the size of the training data that needs to be classified to measure the classification performance of a given solution.

The use of approximate fitness values might seem linked to a reduction of the performance. However, we postulate that for the problem of instance selection (undersampling) in classification problems, using the training data to compute the fitness value is in itself an approximation of how well the solution (the resulting RS) allows us to learn a concept (which may affect how well we can predict the test set). In addition, the search algorithm may end up overfitting the training data. Hence, these are well-known general weaknesses of any existing instance selection algorithm [27].

The main goal of the proposed EUSHC is to drastically reduce the computational cost of EUS and investigate whether this misleads the search or not. To do so, we integrate two different approaches to approximate fitness values: Windowing and a clustering-based surrogate model. The motivation as to why we add the two different components together is given below.

B. Windowing for EUS

The idea of windowing was originally proposed in [17] to accelerate a genetic-based machine learning algorithm. The key idea is to use partial data instead of the entire dataset for each fitness evaluation. This approach begins with splitting the training set into multiple disjoint strata ($W_1, W_2, ..., W_{nw}$). For each generation of the search, each stratum takes a turn to be used in evaluating candidate solutions. Due to the reduction of data quantity at each evaluation, the computational cost decreases accordingly.

In the context of EUS for imbalance classification, windowing was first used in [24]. However, dividing the entire training set into several disjoint windows with equal class distribution produce an important loss of information of the positive class. Therefore, to apply windowing for EUS, minority class sample will be kept to evaluate a chromosome. However, the set of majority class instances is split into several disjoint strata. The size of each subset of majority examples is set to the number of minority class instances to avoid setting a fixed value for the number of strata. Thus, the number of strata is dependant on the imbalanced ratio.

This simple yet effective approach has proven to highly reduce the cost of fitness evaluations without significant loss in classification performance. Although the training data is not classified at once to evaluate one chromosome, during the evolutionary process the algorithm utilises all existing training data. The main drawback of this technique lies in the fact that its reduction of computational cost depends on the imbalanced ratio. For this reason, we will use this technique within the proposed hybrid surrogate model to speed up the fitness computation of only some chromosomes of the population.

C. A hybrid windowing-clustering surrogate model for EUS

Fitness approximation based on surrogate models is an under-explored area in binary/combinatorial optimisation. For

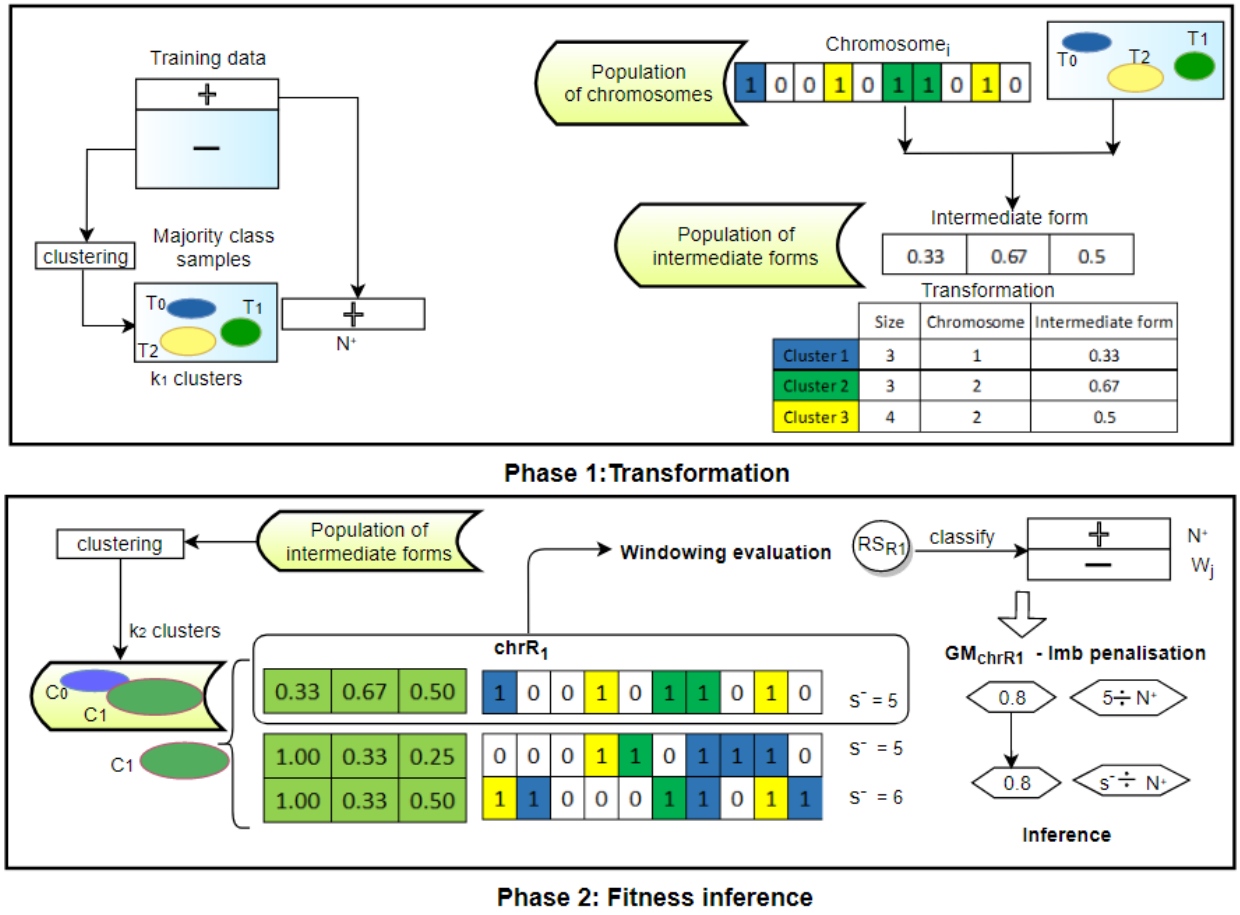

Fig. 1. Workflow of EUSHC: Phase 1 conducts chromosome transformation; in the illustration 1 element out of 3 is selected from T_0 cluster, 2 out of 3 in T_1 cluster, and 2 out of 4 in T_2 cluster. Phase 2 performs fitness inference based on similarities between the transformed chromosomes. Only a representative chromosome from each cluster is evaluated using a windowing approach.

EUS, the main challenge lies at computing distances between different binary chromosomes, so that, fitness values can be either inherited or approximated based on similarity between chromosomes. In [18], we recently proposed a clustering-based surrogate model for EUS (EUSC) that allows us to compute distances between binary chromosomes by transforming them into an real-coding representation. The main advantage of such a model is the ability to very quickly infer the fitness value of a chromosome based on the distance to others without computing any classification. In this work, we extend that approach by hybridising EUSC with a windowing approach. Figure 1 presents the workflow of the entire hybrid model, which consists of the following two phases:

1) Phase 1: Chromosome transformation: The key point of EUSC is related to transforming the binary representation into a real-coding one. This process should be very quick to really take advantage of a surrogate model (rather than using the real fitness evaluation). The main issue with the binary representation is that does not represent well the real phenotype of the chromosome, which is the actual position of the selected instances of the algorithm.

- Step 1: Before starting the evolutionary cycle, all the training samples that belong to the majority class are grouped into k_1 clusters $\{T_0, T_1, ...T_{k_1}\}$. In our experiments, we use the well-known k-means algorithm. The goal of this step is to quickly split the instance space into different regions based on the actual position (i.e. using their feature values) of the majority class examples. Note that this step is the most time-consuming one, but it is only applied once.
- Step 2: During the evolutionary cycle, we will transform binary chromosomes into an intermediate form using the previous clusters. This intermediate forms will have k_1 genes. Firstly, we count the number of selected instances (i.e. genes with a 1) that fall into each of the clusters. Each gene of the intermediate chromosome will be a real value which is computed as the division between the number of selected instances of this cluster and the original number of elements in the cluster. These values produce an intermediate form that tells us approximate information about the location in the instance space of the selected instances.

2) Phase 2: Fitness inference: When binary chromosomes have been transformed into real-coding ones, we can use

similar ideas as implemented in the literature for continuous optimisation problems [28].

- Step 1: The population of chromosomes in their new intermediate form is fed into a clustering algorithm, which splits the different solutions into $k = k_2$ clusters, $C_0, C1, ..., C_{k_2}$. In this way, the clustering task conducted in the intermediate forms will also indirectly separate the chromosomes in the binary space.
- Step 2: Compute the fitness value of only k_2 chromosomes. To accelerate this step, we incorporate here the windowing approach. This means that for those chromosomes a subset of the training set is classified (as describe in the previous subsection) with the RS set. We tested different approaches to decide which chromosomes should be evaluated with the fitness function. In this contribution we pick the centroid chromosome $chrR_i$ from each cluster $\{C_0, ..., C_{k_2}\}$ as a set of representative chromosomes $\{chrR_0, ..., chrR_{k_2}\}$.
- Step 3: Infer the fitness value of the remaining chromosomes. The g-mean values of the $\{chrR_0, ..., chrR_{k_2}\}$ has been calculated in the previous step. To compute the fitness of the rest of the chromosomes, Equation 3 uses the g-mean value GM_{chr_j} of the centroid of the cluster. This means that all the members of a cluster simply inherit the same g-mean value. However, the component of the balance between classes of the fitness function is calculated based on the number of elements selected by the particular solution. We acknowledge that transferring the same g-mean to all members of a cluster may be an oversimplification, and more elaborated solutions could be adopted; however, our experiments show that this simple approach achieves good results.

In the experiments presented in the next section, the above fitness approximation is applied to all fitness evaluations, including the evaluation of the initial population.

IV. Experimental study

This section establishes the experimental set-up (Section IV-A) and discusses the results achieved (Section IV-B).

A. Experimental framework

This empirical study considers 44 two-class imbalanced datasets, commonly used in the literature, from the KEEL dataset repository [29]. All used datasets and their properties are summarised in Table I. For each dataset, the table shows the number of attributes (**Att**), the number of samples (**Samp**), the percentage of examples of each class (**%Class(min,maj)**) and the imbalanced ratio (**IR**). In our experiments we consider a 5-fold cross validation approach, and the averaged g-mean values and runtimes are reported.

To compare the effectiveness of the EUSHC, we will compare the results again three benchmarks: (1) the original EUS, which is expected to be the upper-bound in terms of g-mean, but the slower approach; (2) EUS using windowing to evaluate its fitness function; (3) EUSC, the clustering-based surrogate model without using windowing. Table II summarises the parameters used in the experiments. Note that the NN rule (k=1) has been used as base classifier).

TABLE I
SUMMARY OF DATASETS

Dataset	Att	Samp	%Class(min,maj)	IR
shuttle-c2-vs-c4	9	129	(0.05, 0.95)	20.5
iris0	4	150	(0.33, 0.67)	2.0
glass-0-1-6_vs_5	9	184	(0.05, 0.95)	19.44
glass-0-1-6_vs_2	9	192	(0.09, 0.91)	10.29
glass-0-1-2-3_vs_4-5-6	9	214	(0.24, 0.76)	3.2
glass0	9	214	(0.33, 0.67)	2.06
glass2	9	214	(0.08, 0.92)	11.59
glass4	9	214	(0.06, 0.94)	15.46
glass1	9	214	(0.36, 0.64)	1.82
glass6	9	214	(0.14, 0.86)	6.38
glass5	9	214	(0.04, 0.96)	22.78
new-thyroid1	5	215	(0.16, 0.84)	5.14
new-thyroid2	5	215	(0.16, 0.84)	5.14
ecoli-0_vs_1	7	220	(0.65, 0.35)	0.54
ecoli-0-1-3-7_vs_2-6	7	281	(0.02, 0.98)	39.14
habermanImb	3	306	(0.26, 0.74)	2.78
ecoli1	7	336	(0.23, 0.77)	3.36
ecoli4	7	336	(0.06, 0.94)	15.8
ecoli3	7	336	(0.10, 0.90)	8.6
ecoli2	7	336	(0.15, 0.85)	5.46
yeast-1_vs_7	7	459	(0.07, 0.93)	14.3
page-blocks-1-3_vs_4	10	472	(0.06, 0.94)	15.86
yeast-2_vs_8	8	482	(0.04, 0.96)	23.1
yeast-2_vs_4	8	514	(0.10, 0.90)	9.08
yeast-0-5-6-7-9_vs_4	8	528	(0.10, 0.90)	9.35
wisconsinImb	9	683	(0.35, 0.65)	1.86
yeast-1-4-5-8_vs_7	8	693	(0.04, 0.96)	22.1
abalone9-18	8	731	(0.06, 0.94)	16.4
pimaImb	8	768	(0.35, 0.65)	1.87
vehicle2	18	846	(0.26, 0.74)	2.88
vehicle3	18	846	(0.25, 0.75)	2.99
vehicle0	18	846	(0.24, 0.76)	3.25
vehicle1	18	846	(0.26, 0.74)	2.9
yeast-1-2-8-9_vs_7	8	947	(0.03, 0.97)	30.57
vowel0	13	988	(0.09, 0.91)	9.98
yeast3	8	1484	(0.11, 0.89)	8.1
yeast1	8	1484	(0.29, 0.71)	2.46
yeast4	8	1484	(0.03, 0.97)	28.1
yeast6	8	1484	(0.02, 0.98)	41.4
yeast5	8	1484	(0.03, 0.97)	32.73
shuttle-c0-vs-c4	9	1829	(0.07, 0.93)	13.87
segment0	19	2308	(0.14, 0.86)	6.02
abalone19	8	4174	(0.01, 0.99)	129.44
page-blocks0	10	5472	(0.10, 0.90)	8.79

TABLE II
PARAMETERS USED FOR THE INVOLVED ALGORITHMS.

Algorithm	*Parameters*
EUS	Population Size = 50, Number of Evaluations = 10000, Probability of inclusion HUX = 0.25, Evaluation Measure = g-mean,
EUS_windowing	same as above, but using windowing
EUSC	Same as EUS, plus k_1 = 6, k_2 = 6
EUSHC	same as above, but using windowing

B. Analysis of results

First of all, we compare the runtime required by each one of the methods in every single dataset. Figure 2 plots this comparison. For the sake of clarity, we sort the datasets by the runtime of EUS, and also apply logarithmic scale (base 10) on the vertical axis. We present two subplots, grouping

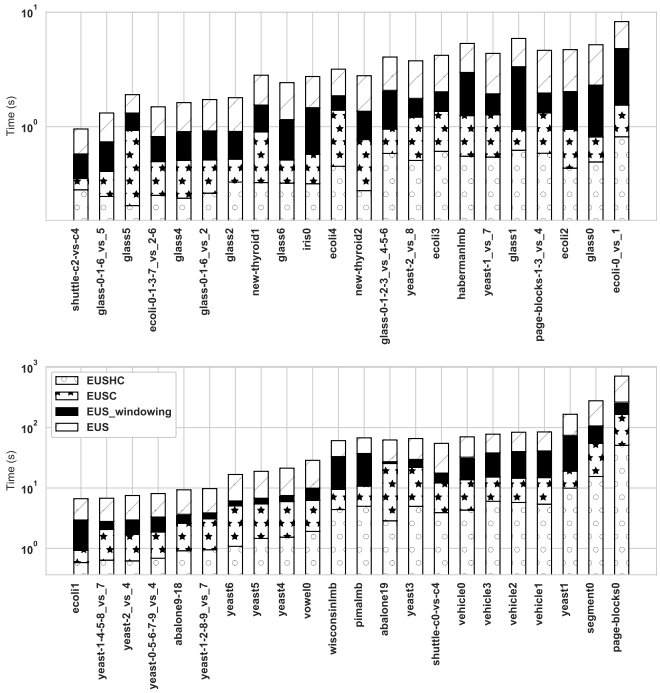

Fig. 2. Comparison of the runtime (in seconds, base 10 logarithmic scale) of the different algorithms over 44 imbalanced datasets, sorted by the runtime of EUS. Runtime of the first 22 datasets (Top), last 22 datasets (Bottom). On average in the 44 datasets, EUS takes 26.44s, EUS_windowing 9.03s, EUSC 6.68s and EUSHC 3.15s.

44 datasets into two halves. Looking at that figure, we can observe that:

- Overall, all the approximation methods consumed an insignificant amount of time to perform undersampling compared to the time demanded by the original EUS.
- Windowing spent a mostly equivalent amount of time to EUS in small datasets, but the time is dramatically reduced in larger ones. As stated above, the windowing approach is also affected by the imbalance ratio.
- Both EUSC and EUSHC show a very low runtime across the 44 examined datasets. As expected the hybrid approach is always faster than EUSC, and it is clearer on larger datasets. On average, EUSHC is roughly 52.84% faster than EUSC (3.15s vs. 6.68s, respectively).

In addition to runtimes, we report the reduction of evaluations provided by the surrogate models. Figure 3 displays a histogram with the total number of evaluations of EUSC and EUSHC compared to the 10000 evaluations performed by EUS for each dataset.

- The two surrogate assisted schemes use a significant lower number of evaluations which is roughly a 20% of the 10000 evaluations used by EUS. The number of real evaluations avoided by the surrogate model varies among datasets. When we use fitness approximation (windowing and/or a surrogate model), the behaviour of the CHC algorithm is also changed. The diversity in the population may be affected, so that, the proportion of chromosomes for which we infer the fitness may be changed in every generation.
  Note that CHC does not necessarily create an offspring of NP elements in every generation. Hence, if the number of chromosomes to be evaluated in a generation is very low (less or equal than k_2), we would not take much advantage of the surrogate model. In the experiments, this effect is more noticeable on datasets with either a very small size or high imbalanced ratio.
- It is important to observe that there is a slight difference in the number of objective function calls between EUSC and EUSHC, and the EUSHC consistently saved more

TABLE III
G-MEAN OBTAINED BY ALL COMPARISON METHODS IN 44 IMBALANCED DATASETS

Dataset	EUS	EUS_windowing	EUSC	EUSHC
shuttle-c2-vs-c4	**0.9577**	0.6449	0.9414	0.7365
iris0	**1.0000**	**1.0000**	**1.0000**	**1.0000**
glass-0-1-6_vs_5	0.9214	0.9151	0.9160	**0.9501**
glass-0-1-6_vs_2	0.6383	0.6164	**0.6651**	0.5815
new-thyroid2	**0.9865**	0.9773	0.9831	0.9746
new-thyroid1	**0.9882**	0.9809	0.9859	0.9653
glass6	0.8889	0.9071	**0.9156**	0.9054
glass5	0.8105	0.9076	**0.9600**	0.9103
glass4	**0.8700**	0.8513	0.8613	0.8531
glass2	0.7194	0.6525	**0.7262**	0.6173
glass1	0.7773	0.7010	**0.7941**	0.7367
glass0	0.8009	0.6176	**0.8047**	0.6595
glass-0-1-2-3_vs_4-5-6	0.9525	0.9385	**0.9647**	0.9546
ecoli-0_vs_1	0.9583	0.9312	0.9581	**0.9615**
ecoli-0-1-3-7_vs_2-6	0.6700	**0.7048**	0.6625	0.6865
habermanImb	0.5475	**0.5635**	0.5521	0.5497
ecoli4	0.8984	0.9362	0.8857	**0.9645**
ecoli3	0.8348	0.8153	**0.8500**	0.8097
ecoli2	0.9000	0.8663	**0.9034**	0.8772
ecoli1	**0.8634**	0.8306	0.8554	0.8424
yeast-1_vs_7	**0.7176**	0.7079	0.7068	0.6669
page-blocks-1-3_vs_4	**0.9674**	0.9399	0.9471	0.9294
yeast-2_vs_8	**0.7931**	0.7496	0.7656	0.7668
yeast-2_vs_4	0.9042	0.8774	**0.9156**	0.8930
yeast-0-5-6-7-9_vs_4	0.7685	0.7663	**0.7901**	0.7535
wisconsinImb	**0.9690**	0.9652	0.9600	0.9590
yeast-1-4-5-8_vs_7	0.6569	0.6088	**0.6604**	0.6149
abalone9-18	**0.7269**	0.6772	0.7224	0.6559
pimaImb	0.6943	0.6749	0.6957	**0.7145**
vehicle0	**0.9164**	0.9027	0.9103	0.9016
vehicle1	0.6729	0.6624	0.6512	**0.6926**
vehicle2	0.9259	0.9175	**0.9265**	0.9173
vehicle3	**0.7280**	0.7142	0.7165	0.7204
yeast-1-2-8-9_vs_7	**0.6721**	0.6078	0.6704	0.6500
vowel0	**0.9897**	0.9719	0.9877	0.9831
yeast3	0.8728	0.8740	**0.8752**	0.8550
yeast1	0.6533	0.6501	**0.6600**	**0.6600**
yeast4	0.8050	0.7799	**0.8288**	0.7970
yeast6	**0.8357**	0.8080	0.8034	0.8031
yeast5	0.9634	0.9494	0.9455	**0.9653**
shuttle-c0-vs-c4	0.9960	**0.9968**	0.9960	0.9960
segment0	**0.9881**	0.9870	0.9876	0.9858
abalone19	0.6258	0.6061	**0.7214**	0.6556
page-blocks0	**0.9117**	0.9038	0.9096	0.9085
Wins	**18**	**4**	**18**	**8**

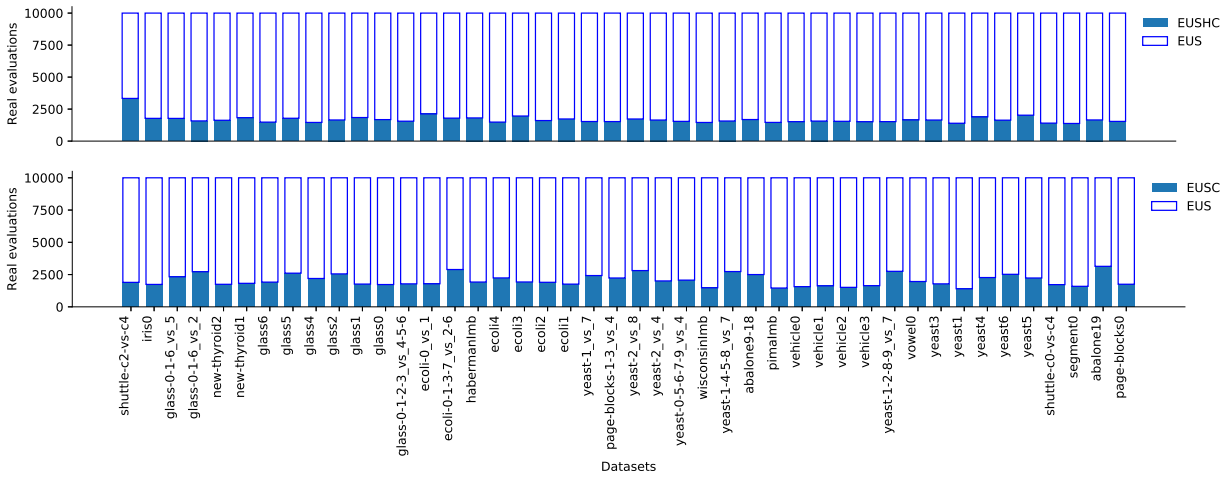

Fig. 3. The number of fitness function calls in the original EUS, EUSC and EUSHC

real evaluations in most of the datasets. This behaviour may seem unexpected as both methods are using a clustering-based surrogate model to reduce the number of evaluations, so that, both should report a similar number. However, the hybrid approach introduces windowing that changes the behaviour of EUS, and the search. Including windowing may also affect the quality of the chromosomes and, as explained above, the number of chromosomes to be evaluated in each generation.

Until now, it is clear the advantage of the fitness approximation approaches with respect to EUS in terms of efficiency. However, reducing the runtime would not be of any value if the classification performance is massively deteriorated. Table III shows the average g-mean performance of all the algorithms in test data. Values in bold indicate that the algorithm at the column achieves the highest g-mean in the dataset at the row. Additionally, an extra row at the end displays the number of times that each algorithm wins over 44 datasets. We also compare the number of wins, ties and losses of EUSHC against each reference undersampling algorithm, displayed in Figure 4. Looking at the above table and figure, we can observe that:

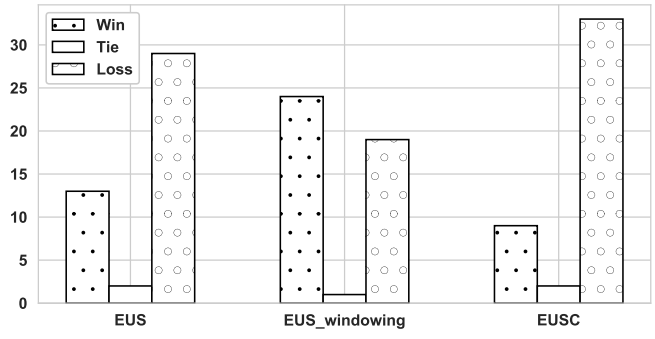

Fig. 4. Comparison of EUSHC and reference undersampling algorithms with respect to the number of wins, ties, and losses over 44 imbalanced datasets.

- Despite using about 80% more evaluations, EUS does not always provide the best classification performance. This result shows that the use of approximations may result in even better results, reducing overfitting of the training set. As stated in Section III-A, using the training data is already an approximation of how well this data represents the concept to be learned.
- In this experiment, we can highlight EUSC, which seems to be very competitive with respect to EUS, obtaining the same number of wins out of the 44 datasets. EUSHC also finds the best solution in 8 out of 44 datasets.
- Over 44 datasets, EUSHC shows a greater number of wins with respect to EUS_windowing. It is predicted that EUSHC loses EUS and EUSC frequently as it applies two stages of approximation. However, figures in Table III show a very reduced difference in g-mean between our hybrid approach and the other algorithms. Note that the difference in g-mean is only more noticeable in those datasets either having high IR or low number of samples with high IR.

In summary, the proposed EUSHC highly reduces the computational cost of the EUS algorithm (about an 88.08% on average - from 26.44s to 3.15s), and the classification performance seems comparable to EUS and the other approximation methods. Looking at all the results presented in this contribution, when the number of instances is low, it is reasonable not to use a surrogate or windowing approach as the original EUS will not suffer from a high computational cost. However, in larger datasets, the benefits of the proposed approach are promising.

V. Conclusion

In this paper we have presented a hybrid surrogate model to accelerate evolutionary undersampling for imbalanced classification problems. The proposed approach approximates fitness values of the chromosomes using a clustering-based surrogate model together with a windowing approach. The

entire search is guided by approximate fitness values aiming to highly reduce the computational cost. In our experiments in 44 standard imbalanced datasets we show that we can highly reduce the runtime required to perform evolutionary undersampling, especially in larger datasets, without incurring in a noticeable classification performance loss. As such, the proposed approach contributes towards the design of fitness approximation models based on surrogate models in evolutionary algorithms for instance selection/undersampling. As future work, we plan to incorporate the proposed approach to big data frameworks and analyse the effect of fitness approximation in bigger datasets.

REFERENCES

[1] S. García, J. Luengo, and F. Herrera, *Data preprocessing in data mining*. Springer, 2015.

[2] I. Triguero, D. García-Gil, J. Maillo, J. Luengo, S. García, and F. Herrera, "Transforming big data into smart data: An insight on the use of the k-nearest neighbors algorithm to obtain quality data," *WIREs Data Mining and Knowledge Discovery*, vol. 9, no. 2, p. e1289, 2019.

[3] H. Song, I. Triguero, and E. Ozcan, "A review on the self and dual interactions between machine learning and optimisation." *Prog Artif Intell.*, vol. 8, p. 143165, 2019.

[4] B. Xue, M. Zhang, W. N. Browne, and X. Yao, "A survey on evolutionary computation approaches to feature selection," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 4, pp. 606–626, Aug 2016.

[5] J. R. Cano, F. Herrera, and M. Lozano, "Using evolutionary algorithms as instance selection for data reduction in kdd: an experimental study," *IEEE Transactions on Evolutionary Computation*, vol. 7, no. 6, pp. 561–575, Dec 2003.

[6] V. López, A. Fernández, S. García, V. Palade, and F. Herrera, "An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics," *Information sciences*, vol. 250, pp. 113–141, 2013.

[7] B. Krawczyk, "Learning from imbalanced data: open challenges and future directions," *Progress in Artificial Intelligence*, vol. 5, no. 4, pp. 221–232, 2016.

[8] M. Galar, A. Fernández, E. Barrenechea, and F. Herrera, "EUSBoost: Enhancing ensembles for highly imbalanced data-sets by evolutionary undersampling," *Pattern Recognition*, vol. 46, no. 12, pp. 3460–3471, 2013.

[9] S. García and F. Herrera, "Evolutionary undersampling for classification with imbalanced datasets: Proposals and taxonomy," *Evolutionary computation*, vol. 17, no. 3, pp. 275–306, 2009.

[10] L. J. Eshelman, "The chc adaptive search algorithm: How to have safe search when engaging in nontraditional genetic recombination," vol. 1, pp. 265–283, 1991.

[11] I. Triguero, M. Galar, H. Bustince, and F. Herrera, "A first attempt on global evolutionary undersampling for imbalanced big data," in *2017 IEEE Congress on Evolutionary Computation (CEC)*, June 2017, pp. 2054–2061.

[12] Y. Jin, "A comprehensive survey of fitness approximation in evolutionary computation," *Soft computing*, vol. 9, no. 1, pp. 3–12, 2005.

[13] Y. Sun, H. Wang, B. Xue, Y. Jin, G. G. Yen, and M. Zhang, "Surrogate-assisted evolutionary deep learning using an end-to-end random forest-based performance predictor," *IEEE Transactions on Evolutionary Computation*, pp. 1–1, 2019.

[14] Y. Jin, "Surrogate-assisted evolutionary computation: Recent advances and future challenges," *Swarm and Evolutionary Computation*, vol. 1, no. 2, pp. 61–70, 2011.

[15] T. Bartz-Beielstein and M. Zaefferer, "Model-based methods for continuous and discrete global optimization," *Applied Soft Computing*, vol. 55, pp. 154–167, 2017.

[16] J. R. Cano, F. Herrera, and M. Lozano, "Stratification for scaling up evolutionary prototype selection," *Pattern Recognition Letters*, vol. 26, no. 7, pp. 953–963, 2005.

[17] J. Bacardit, D. E. Goldberg, M. V. Butz, X. Llorà, and J. M. Garrell, "Speeding-up pittsburgh learning classifier systems: Modeling time and accuracy," in *International Conference on Parallel Problem Solving from Nature*. Springer, 2004, pp. 1021–1031.

[18] H. Lam Le, D. Landa-Silva, M. Galar, S. García, and I. Triguero, "EUSC: A clustering-based surrogate model to accelerate evolutionary undersampling in imbalanced classification," *Arxiv*, 2020, to appear.

[19] G. E. Batista, R. C. Prati, and M. C. Monard, "A study of the behavior of several methods for balancing machine learning training data," *ACM SIGKDD explorations newsletter*, vol. 6, no. 1, pp. 20–29, 2004.

[20] S.-H. Oh, "Error back-propagation algorithm for classification of imbalanced data," *Neurocomputing*, vol. 74, no. 6, pp. 1058–1061, 2011.

[21] B. Krawczyk, M. Woźniak, and G. Schaefer, "Cost-sensitive decision tree ensembles for effective imbalanced classification," *Applied Soft Computing*, vol. 14, pp. 554–562, 2014.

[22] M. Galar, A. Fernandez, E. Barrenechea, H. Bustince, and F. Herrera, "A review on ensembles for the class imbalance problem: bagging-, boosting-, and hybrid-based approaches," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 42, no. 4, pp. 463–484, 2011.

[23] E. Ramentol, Y. Caballero, R. Bello, and F. Herrera, "SMOTE-RSB*: a hybrid preprocessing approach based on oversampling and undersampling for high imbalanced data-sets using smote and rough sets theory," *Knowledge and information systems*, vol. 33, no. 2, pp. 245–265, 2012.

[24] I. Triguero, M. Galar, S. Vluymans, C. Cornelis, H. Bustince, F. Herrera, and Y. Saeys, "Evolutionary undersampling for imbalanced big data classification," in *2015 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2015, pp. 715–722.

[25] T. M. Cover, P. Hart *et al.*, "Nearest neighbor pattern classification," *IEEE transactions on information theory*, vol. 13, no. 1, pp. 21–27, 1967.

[26] A. C. Martínez-Estudillo, C. Hervás-Martínez, F. J. Martínez-Estudillo, and N. García-Pedrajas, "Hybridization of evolutionary algorithms and local search by means of a clustering method," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 36, no. 3, pp. 534–545, 2005.

[27] S. Garcia, J. Derrac, J. R. Cano, and F. Herrera, "Prototype selection for nearest neighbor classification: Taxonomy and empirical study," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, no. 3, pp. 417–435, 2011.

[28] H.-S. Kim and S.-B. Cho, "An efficient genetic algorithm with less fitness evaluation by clustering," in *Evolutionary Computation, 2001. Proceedings of the 2001 Congress on*, vol. 2. IEEE, 2001, pp. 887–894.

[29] I. Triguero, S. González, J. M. Moyano, S. García López, J. Alcalá Fernández, J. Luengo Martín, A. Fernández Hilario, J. Díaz, L. Sánchez, F. Herrera *et al.*, "Keel 3.0: an open source software for multi-stage analysis in data mining," 2017.