One-Class Classification for Selecting Synthetic Datasets in Meta-Learning

1st Regina R. Parente
Centro de Informática (CIn)
Universidade Federal de Pernambuco (UFPE)
Recife, Brazil
rrp@cin.ufpe.br

2nd Ricardo B. C. Prudêncio
Centro de Informática (CIn)
Universidade Federal de Pernambuco (UFPE)
Recife, Brazil
rbcp@cin.ufpe.br

***Abstract*—Algorithm selection is a challenging task in machine learning. Meta-learning treats algorithm selection as a supervised learning task, in which training examples (i.e., meta-examples) are generated from experiments performed with a set of candidate algorithms in several datasets. The small availability of real datasets in some domains can make it difficult to generate good sets of meta-examples. An alternative is the use of synthetic datasets. Unfortunately, not all synthetic datasets can be considered equally relevant and representative compared to real datasets. Thus simply adopting a high number of arbitrary synthetic datasets increases the computational cost of performing experiments, without necessarily improving the quality of meta-learning. In this paper, we treat the selection of relevant synthetic datasets for meta-learning as an One-Class Classification (OCC) problem. In this problem, it is assumed the availability of instances associated to a single class of interest (the positive class) and a large set of unlabelled instances (the unknown class). The objective is to classify which unlabelled instances most likely belong to the positive class. In our context, OCC techniques are used to select the most relevant synthetic datasets (unknown class), by considering the real datasets (positive class) available. In our work, we conducted experiments in a case study in which we adopted a data manipulation procedure to produce synthetic datasets and two OCC techniques for dataset selection. The results revealed that it was actually possible to select a reduced number of synthetic datasets while maintaining or even increasing meta-learning performance.**

***Index Terms*—meta-learning, algorithm selection, one-class classification**

I. Introduction

It is a commonplace that we are currently facing a special moment of massive application of machine learning in different domains of science and industry. The increasing interest demands for solutions to support the design of learning systems and to address relevant issues like which algorithms to use, how to tune them and how to process training data. Such solutions are crucial specially for non-expert users of machine learning techniques.

This paper is focused on the algorithm selection problem, which has been successfully dealt with using meta-learning [1]. In this solution, algorithm selection models are built by acquiring knowledge from benchmarking experiments in different datasets. In meta-learning, each training example (i.e., a meta-example) is associated to a dataset of interest (e.g., a classification task) and usually stores: (1) the dataset's characteristics (the meta-attributes), such as number of examples, number of instances, correlations between attributes, among others; (2) a meta-label, indicating the best algorithm for that dataset, assigned from empirical evaluation of a set of candidate algorithms. Based on a set of meta-examples generated from several problems, a meta-learner is constructed for selecting the best algorithm for new datasets based on their characteristics. The interest in meta-learning has increased along the years due to its broad applicability (e.g., for hyperparameter optimization and workflow design) [1].

The quality of a meta-learner, as in any other machine learning model, depends on the quantity and quality of the training examples. In meta-learning, each meta-example is constructed from a dataset, usually stored in benchmark repositories. The difficulty in this context is that the number of datasets available in repositories is typically limited and thus the number of meta-examples produced is not always suitable for obtaining a good meta-learner [1]. To minimize this difficulty, synthesized datasets can be used, in combination with real datasets, to increase the number of meta-examples [2]. The use of synthesized data, however, may generate another difficulty: many of the synthetic datasets may be irrelevant when one considers the characteristics of real problems. The distribution of meta-attributes observed in the synthetic examples may be quite different compared to real problems. Selecting only the most relevant synthetic datasets would save computational costs of performing experiments at the same time maintaining (or even improving) meta-learning performance.

In our work, we propose a novel solution for selecting relevant synthetic datasets for meta-learning, by adopting One-Class Classification (OCC) techniques [3]. In OCC, it is assumed the availability of instances belonging to a class of interest (the positive or target set), while the other instances are unlabelled (referred to as the unknown set). OCC is applicable in scenarios in which instances in the negative class are difficult to collect or are costly to label. The selection of synthetic datasets for meta-learning is directly treated in our work as an OCC task. The real available datasets available (e.g., benchmark datasets in repositories) are considered as representatives of relevant problems, thus forming the positive set. Synthetic datasets, in turn, form the unknown set of instances. In fact, although it is relatively easy to produce a

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

large base of synthetic datasets, it is not known a priori which ones can be safely adopted as relevant benchmark problems in experiments. In our proposal, an OCC technique receives as input the positive set of meta-examples and is then used to select the relevant synthetic datasets in the unknown class.

In order to test the feasibility of the proposal, we carried out a case study in which a data manipulation technique [4] was used to generate 983 synthesized datasets from 64 real datasets and two OCC techniques were adopted for dataset selection: the Positive and Unlabelled learning (PU learning) proposed in [5] and the Nearest Neighbor Data Description (NNDD) [6]. In the experiments, algorithm selection performance was actually improved when the synthetic datasets were adopted to produce meta-examples. Additionally, the proposed approach was able to reduce the number of meta-examples (about 32% and 62% respectively for PU and NNDD), while maintaining or even increasing meta-learning performance obtained when all synthetic datasets were adopted.

Section II presents a brief overview of meta-learning applied to algorithm selection, while Section III presents the OCC problem. Section IV describes the proposal and implementation. Section V presents the experimental results. Finally, Section 6 presents the general remarks and future work.

II. Meta-Learning for Algorithm Selection

Meta-learning aims to improve learning performance by exploiting knowledge acquired from experience on different applications and case studies [1]. This is a broad objective. In our work, we focus on meta-learning for algorithm selection by investigating how to relate learning algorithms and the features of problems which they are applied to [7]. The central defiance, however, consists of choosing a particular method or finding out which tool will be used for a new classification problem, represented by a specific dataset, in such a way that the solutions presented are adequate and efficient. Each problem has its peculiarities, considering the nature of the dataset represented.

Fig. 1 illustrates the general meta-learning process for algorithm selection. The process begins with the acquisition of a repository of datasets that appropriately represents the application domain in which the algorithm recommendation will be made. The next steps are to evaluate the candidate algorithms on each problem and to extract the dataset characteristics, according to predefined descriptive features (the meta-attributes). For each dataset, a meta-example is generated, formed by the meta-features and the meta-class[1], respectively. The set of meta-examples available is called meta-data. In order to induce the mapping between the input meta-features and the meta-class, a supervised classification algorithm is applied to produce a meta-learner (i.e., a classifier that associates meta-features to the best algorithms). Through this, it is possible to use the meta-learner to recommend algorithms for new problems. In the last years, concepts and techniques of meta-learning have been generally applied in various application domains, such as selection of models of temporal series prediction [8], software engineering [9], bioinformatics [10] and characterization measures of ensemble systems [11].

[1]Label indicating the best algorithm for the problem.

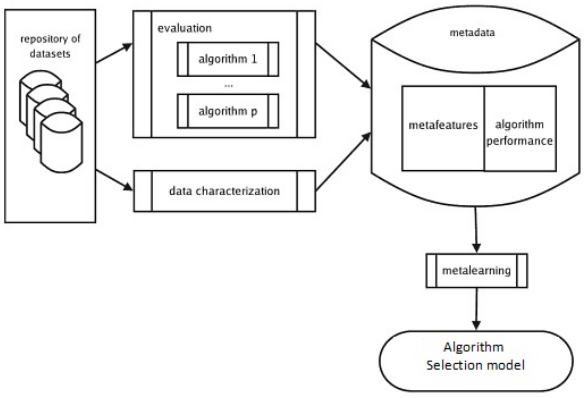

Fig. 1. Meta-Learning process for algorithm selection. Adapted from [1]

The generation of meta-examples is an important issue in meta-learning since its success strongly depends on a sufficient number of problem examples that are at the same time relevant and not redundant. Relying on a repository of datasets like UCI [12] may be the first alternative to be considered. Although intuitive, depending on the class of problem, it is not easy to find a large number of problem examples to generate meaningful meta-data to obtain a reliable model. In the literature, approaches are found to circumvent the aforementioned question, such as the generation of synthetic problems [4], [13].

III. One-Class Classification

In traditional two-class or multi-class classification, training examples for all classes are required. However, in many real problems, it may be difficult or expensive to obtain examples of one or more classes [14]. One-Class Classification (OCC) or unary classification [3] aims to build a classification model from training examples of a positive class (or target class), such that it accepts as many instances as possible from the positive class at the same time minimizing the chance of accepting non-positive instances. In OCC, examples of non-target classes are missing or available in limited quantity. A classification model can be built using only labelled positive data, making unary classification more difficult than the conventional classification problem.

Recently, there has been a considerable amount of research carried out in the field of OCC [15]–[17]. A difficulty that can be pointed out in OCC is to decide which attributes should be used to find the best separation of positive and non-positive class examples. In addition, particularly when the class boundaries of the data are complex, the required number of training objects might be very high.

Regarding the availability of labelled negative instances, three major categories are studied in OCC [3]: (1) Learning

with positive examples only [18], [19]; (2) Learning with some negative examples of artificially generated samples and with positive examples [20]; and (3) learning with positive and unlabelled data [21], [22]. In general, the main goal behind these strategies is to build a decision boundary around the positive data so as to differentiate the unknown negative instances (or unknown class) from the positive data.

In this work, the selection of synthetic datasets is treated as a one-class learning problem. The present proposal nevertheless is related to category (3), since the real datasets represent the instances of the positive class, while the synthetic data represent the unlabelled data. For more information about OCC, the reader is referred to [3] and references cited therein.

IV. One-Class Classification for Selection of Synthetic Datasets

Data generation can be used to increase the number of datasets available for meta-learning, resulting in a potential positive impact in performance in the meta-level. However, although obtaining relevant meta-examples in the process of dataset generation, it is also possible to obtain redundant or outlier meta-examples [4]. Additionally, as stated in Section II, in meta-learning, the labelling process of meta-examples is performed through empirical evaluation of candidate algorithms, which can be computationally expensive. Therefore, selecting relevant datasets for meta-example generation has a twofold motivation: improving the quality of the set of meta-examples itself and reducing the computational cost of experiments performed to label meta-examples. Additionally, in the literature, there is a scarcity of works related to the selection of datasets used to generate meta-examples. In general, the focus is on the generation and selection of meta-features.

Given such motivation, in the current work, we investigate how to improve the construction of meta-example sets by combining data manipulation for datasets generation [4] and OCC techniques for meta-examples selection. In this proposal, the selection of synthetic datasets is treated as a one-class learning problem, as mentioned in Section III.

The meta-feature distributions for real and synthetic datasets may be different since not all synthetic datasets are representative of real ones. Selecting only the most relevant synthetic datasets may be a good strategy for producing meta-examples. In this context, the meta-examples generated from real datasets belong to the positive class, while the meta-examples generated from synthetic datasets belong to the unknown set (i.e., some of them can be positive/relevant while others can be irrelevant). Our hypothesis is that this problem can be adequately dealt with OCC techniques.

The selection of meta-examples in [2], [23], were performed adopting an uncertainty sampling method for active learning. The results reported in [23] were improved in [2] when the k-Nearest Neighbors (k-NN) algorithm was adopted as a meta-learner. Although previous works have achieved promising results, identified limitations, such as the fact that the uncertainty sampling method for active learning was adopted in both studies also motivate our investigation with OCC techniques.

Fig. 2 presents the proposal developed in the current work. First of all, a collection of real datasets is given as input for a *Data Manipulation* module. Then this module generates synthetic datasets from the original datasets. Next, all datasets (original and synthetic) are given as input for the *Data Characterization* module, in order to compute the meta-features of each dataset, resulting in a large set of unlabelled meta-examples. In the following step, the *One-Class Classification* module is adopted to select the most relevant meta-examples in the unknown set. The candidate algorithms are then evaluated after an empirical evaluation procedure to assign the meta-class label of the selected meta-examples (*Algorithm Evaluation module*), therefore forming the meta-base of examples (*Meta-Data*). Finally, the meta-data serves as input for the *Meta-learning* module, which has as output an algorithm selection model. As follows, the details of the implementation of each module is provided in a case study performed to test the viability of the proposal.

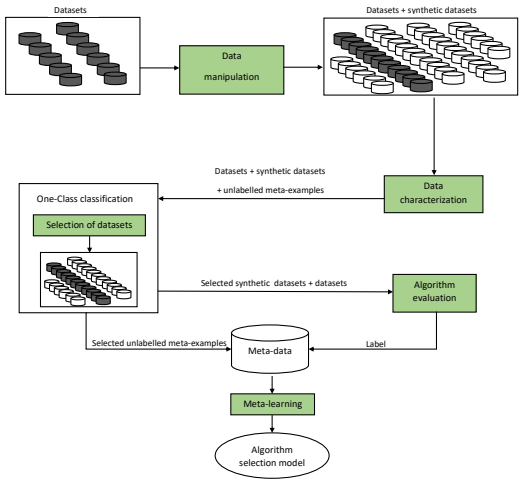

Fig. 2. Architecture of the proposed solution.

A. Data Manipulation

In this work, we adopt a method for data manipulation called *datasetoids*, proposed in [4]. This approach, used for augmenting the number of meta-examples, implies the manipulation of existing datasets. Different manipulation operators can be adopted to produce new synthetic datasets. In [4], a simple operator is proposed. A synthetic dataset is generated from a determined real dataset by switching the target attribute with an independent categorical attribute. In this way, the target attribute of the original dataset becomes an attribute in the datasetoid, and the selected independent attribute becomes the target attribute. In order to generate synthetic datasets for classification, the process is repeated for every categorical attribute of the dataset, creating as many new datasets as there are categorical attributes existing in the dataset. In this case, the dataset must contain at least one

categorical attribute. Fig. 3 illustrates the process of generation of two classification new synthetic datasets (datasetoids) from a dataset with two symbolic attributes.

The original sets of problems used in this work are classification sets collected in UCI [12] repository. From 64 original problems collected in these repositories, 983 new manipulated datasets were produced.

B. Data Characterization

In this module, original and synthetic datasets are described by a set of 2 characterization measures: the class entropy and the average entropy of the attributes [24]. These measures are expected to contain relevant information on the behavior of decision trees since this algorithm utilizes the concept of entropy. Although there are certainly other measures (i.e., meta-features) that could collaborate to improve the meta-learning results, the use of previously adopted meta-features enables us to direct our attention on the combination of OCC techniques and synthetic datasets. The characterization will therefore result in an unlabelled meta-example for each dataset considered in the case study. Both (original and synthetic datasets) jointly with their respective meta-examples are supplied as input to the *One-Class Classification* module.

C. One-Class Classification

In one-class learning, two instance sets are usually available: (1) a set $\mathcal{P}$ of examples in the positive class (target class); and (2) a set $\mathcal{U}$ of examples with unknown class. The $\mathcal{U}$ set can have both examples belonging to the positive and the negative class. A classifier is constructed to select the unknown instances with a higher probability of belonging to the positive class. In our proposal, the positive class represents the original datasets and the unknown instances are the synthetic datasets $\mathcal{U}$. OCC learning is therefore performed to classify the synthetic datasets that are more related to the real ones.

In our implementation, we considered two different OCC methods: *PU learning* proposed in [5] and Nearest Neighbor Data description (NNDD) [6].

1) PU Learning: We considered the OCC method *PU learning*, which learns a classifier to select the most likely positive and negative instances in the unknown set [15]. This method is composed by two steps. The first one is adopted to produce an initial training set for building the classifier. The second step is the learning process itself, in which instances are included in the training set.

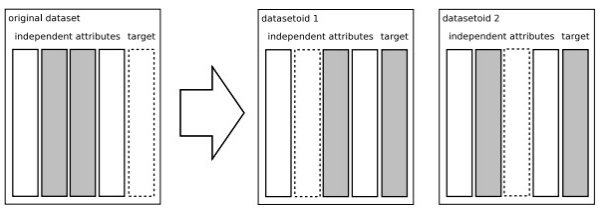

Fig. 3. Illustration of classification datasetoids created from a dataset with two symbolic attributes. Reproduced from [4]

Algorithm 1 illustrates the first step of the method. Initially, a small set $\mathcal{S}$ of examples of positive class $\mathcal{P}$ (named as *spies*) is mixed with the unlabelled data $\mathcal{U}$. As suggested in [15], we used 10% of $\mathcal{P}$, selected randomly, as spies. Datasets $\mathcal{P} - \mathcal{S}$ and $\mathcal{U} \cup \mathcal{S}$ are temporarily labelled as the positive and negative set, respectively, and the *Naive Bayes* (NB) classifier is built. An instance in $\mathcal{U}$ is finally assumed as negative if its positive class probability does not exceed a predefined threshold b. Finally it returns, the subset N_r of the most likely negative instances in the unknown set. This is an initial classification that will be refined in the second step of PU.

Algorithm 1: PU Learning - Initialization of the Training Set

```
N_r = NULL;
U_r = NULL;
S = sample(P,10%);
assign each instance in P − S to the positive class;
assign each instance in U ∪ S to the negative class;
build a NB model g by using P − S and U ∪ S as
  training set;
classify each instance in S using g and determine
  threshold b;
for each instance e ∈ U do
    use model g to compute P(positive|e)
    if probability P(positive|e) < b then
        N_r = N_r ∪ {e};
U_r = U − N_r;
return N_r and U_r for the second step.
```

An important issue in the first step is the decision threshold b adopted to assign the negative instances. A straightforward idea is to define b as the minimum positive class probability observed among the instances in $\mathcal{S}$. Nevertheless, it is also expected the presence of noise in the positive set, in such a way the minimum probability is not reliable [15]. For example, the class probability of an outlier instance in $\mathcal{S}$ could be very close to 0 or much smaller than most of all negative instances. This is also true in our context since not all datasets in repositories like UCI can be undoubtedly assumed as relevant. In our work, we will define a noise level $l\%$ of examples, as a parameter of the method. Then, we order the examples in $\mathcal{S}$ by their positive class probabilities and set the threshold b in such a way that $l\%$ of the positive examples has class probability lower than b. In our experiments, we evaluated the behavior of the PU Learning method with various noise levels.

In the second step of the method (**Algorithm 2**), a *Support Vector Machine* (SVM) is constructed. $\mathcal{P}$ and N_r are used as training sets to learn a SVM classifier. To minimize the effects of unbalanced data on training, we use oversampling, in which examples of minority classes are randomly replicated. The SVM is applied to classify the examples in U_r. The examples classified as positive (U_p) are returned as the set of relevant

examples and the negative examples ones (V_n) are discarded. We use the implementation available in the WEKA framework [25] and the default parameters of this implementation.

Algorithm 2: PU Learning - Classification of Instances in the Unknown Set

U_p = NULL;
Oversampling($\mathcal{P}$,N_r);
use $\mathcal{P}$ and N_r to train a SVM classifier;
classify U_r by using the SVM model;
let $V_n \subset U_r$ be the set of instances the current model classified as negative;
let $U_p \subset U_r$ be the set of instances the current model classified as positive;
return U_p

2) Nearest Neighbor Data Description (NNDD): The second OCC method is based on the Nearest Neighbor classifier (NN) [26]. NNDD is trained only with the positive class, in our case composed by the meta-examples produced from real datasets. In the test phase, an unknown instance (datasetoid) is defined as relevant by NNDD by considering the following procedure. First, given an example $e \in \mathcal{U}$, NNDD finds its nearest neighbor in the positive class $NN_{tr}(e) \in \mathcal{P}$. Second, the nearest positive neighbor of $NN_{tr}(e)$ is returned, i.e., $NN_{tr}(NN_{tr}(e))$. Finally, the unknown example e is considered relevant if the distance to its nearest positive neighbor $dist(e, NN_{tr}(e))$ is smaller than the distance between $NN_{tr}(e)$ and its own nearest positive neighbor. This is summarized in Eq. 1, in which $dist$ is the euclidean distance between two meta-examples:

$$\frac{dist(e, NN_{tr}(e))}{dist(NN_{tr}(e), NN_{tr}(NN_{tr}(e)))} \leq 1 \qquad (1)$$

Otherwise, e is classified as irrelevant. Fig. 4 illustrates a case where a synthetic dataset is accepted by NNDD since $\frac{d1}{d2} \leq 1$.

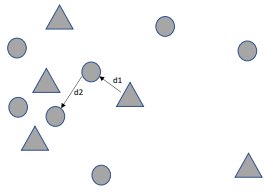

Fig. 4. Example of synthetic dataset accepted by NNDD. The synthetic dataset is very close to a real dataset. Real datasets are represented as circles and unknown instances are represented as triangles.

In our proposal, the idea behind is to select as many relevant synthetic datasets as possible for each original instance (datasets). For each original instance, a maximum of 6 synthetic instances was evaluated, in other words, k = 6. The synthetic datasets accepted by NNDD will be labelled by the *Evaluation* module, described in the next subsection.

D. Algorithm Evaluation

In this investigation, at the base level, we choose the same meta-learning task originally used to evaluate the datasetoids approach in [4]. There are three class labels of meta-examples, p, u or t: the winner is the pruned tree, the unpruned tree or the one that is tied, respectively. The algorithm selection problem in this work consists of predicting, a priori, if pruning a decision tree will enhance the quality of the model or not, given a determined problem. The performance measure used at the base level was the accuracy of classification, estimated using 10-fold cross-validation. Each meta-example represents one problem (i.e., an original dataset or a synthetic dataset). The class of each meta-example is based on the results of the experiments on the corresponding problem. For more information about the implementations, the reader is referred to [4] and references cited therein.

Table I presents the classes distribution for the meta-data, both the original and synthetic data. As it can be seen, the class distributions are different, which indicate that there may be some underlying differences between real and synthetic datasets. However, the differences are not so large in the class distribution considering the real and synthetic datasets separately.

TABLE I
CLASS DISTRIBUTION (%) OF THE META-DATA CORRESPONDING TO THE REAL AND SYNTHETIC DATASETS

Meta-data	No. of meta-examples	Pruned tree (p)	Unpruned tree (u)	Tie (t)
Real	64	36	23	41
Synthetic	983	37	10	53

E. Meta-Learning

The result of the previous step is a labelled meta-examples set. Each meta-example is associated to a dataset and it stores the descriptive characteristics and the class value indicating the best candidate algorithm for the dataset. In the current work, we adopt two classification algorithms as meta-learner: *Random Forest* (RF) and k-NN. These algorithms are used as meta-classifiers which will indicate the best model for a determined test dataset. In our prototype, we use the implementations developed in Java programming language available on WEKA environment [25].

V. EXPERIMENTS

In this section, we present the experiments to evaluate the quality of the trained meta-learners with the data selected using one-class learning techniques.

A. Evaluation Methodology

In order to evaluate the learning performance at the meta-level, we adopted a holdout evaluation with 500 repetitions. In each repetition, the repository of real datasets is randomly partitioned into two mutually exclusive partitions (test and training), where 10% of the datasets are used for testing. As

the training partition we use the 90% of the remaining real datasets and all synthetic datasets generated from them. Hence, we certify that the synthetic datasets generated from the test datasets are not included in the training partition.

In each holdout iteration, the OCC technique (either PU or NNDD) is applied to select synthetic datasets in the training partition. For the NNDD method, up to 300 synthetic instances have been selected and labelled. The meta-learner is then constructed using the real and the selected synthetic datasets for training and it is evaluated for the algorithm selection in the test datasets. The performance of meta-learning is the average accuracy over the 500 executions of holdout. In order to evaluate the impact of the noise level, we repeated this methodology by varying the expected noise level in the positive examples (parameter b).

The proposed approach is compared with two baselines: (1) using as training data only the original datasets (OD); (2) using the original sets together with all the synthetic datasets generated from these sets without selection (OD + all synthetics). These baselines correspond to extreme situations in which either we do not consider any synthetic dataset for meta-learning or we consider all synthetic datasets without any selection of relevant ones.

B. Results

In this paper, the results are presented according to the adopted OCC techniques, which are PU learning and NNDD. The proposed approach is compared with two baselines, as described in Section V-A.

The average meta-learning accuracy obtained by the baseline (OD + all synthetics) was 60.14%, which was significantly higher than using only the original datasets (average accuracy of 46.20%). However, a large number of datasets is adopted, which increases the cost of evaluating candidate algorithms in the meta-example labelling process. So the objective of OCC learning is to select a reduced number of relevant synthetic datasets to achieve an accuracy level close to using all datasets (60.14%). The results obtained after the selection of each of the adopted techniques are presented below.

1) PU learning: As described in Section IV-C1, the dataset selection in PU Learning is performed into two steps. In the first step, a set of negative instances is selected for training the SVM classifier in the second step. The number of selected negative instances depends on the noise level considered (see Fig. 5). Fig. 6 and Fig. 7 show the performance of the RF and k-NN meta-learners, respectively, by varying the noise level. Initially, we consider two extreme cases:

- For extremely low noise levels, a very low number of negative instances is selected for the second step, compared to the number of positive examples. As a consequence, the trained SVM tended to classify a high number of the remaining synthetic datasets as positive (many of them are possibly non-relevant). The meta-learning accuracy, in this case, is sub-optimal and lied between the two baselines (the initial segment of Fig. 6 and Fig. 7);
- In turn, for extremely high values of noise level, an excessive number of negative instances will be given for SVM learning in the second step and then only few synthetic datasets will be finally classified as positive (relevant). Meta-learning accuracy, in this case, approaches the accuracy value observed by using only the original datasets (the final segment of Fig. 6 and Fig. 7).

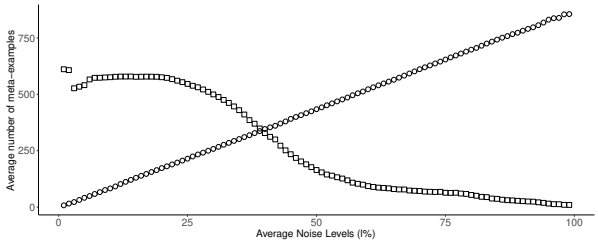

Fig. 5. The relation between the number of selected negative instances (N_r) in the first step considered truly negative (circles) and the number of selected instances (U_p) in the second step of PU learning considered relevant, i.e., belonging to the positive class(squares).

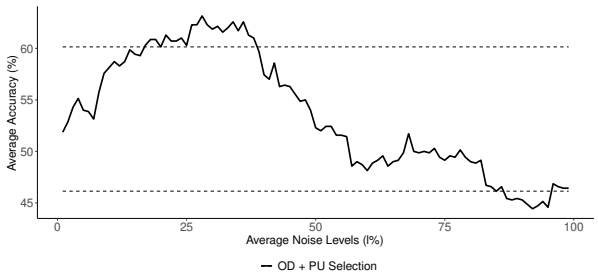

Fig. 6. Average accuracy obtained by RF using PU learning for selecting synthetic datasets. The training set is initialized with all the original datasets, leaving only synthetic datasets for PU selection. Dashed horizontal lines represent the accuracy obtained using all original datasets and synthetic datasets (top) or just the original datasets (bottom).

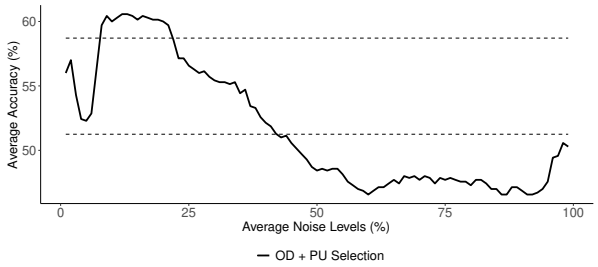

Fig. 7. Average accuracy obtained by k-NN using PU learning for selecting synthetic datasets. The training set is initialized with all the original datasets, leaving only synthetic datasets for PU selection. Dashed horizontal lines represent the accuracy obtained using all original datasets and synthetic datasets (top) or just the original datasets (bottom).

By considering the RF meta-learner, for intermediate values of noise level (from 17% to 38%), the observed meta-learning accuracy was higher than using all synthetic data. The best result was 63.14% accuracy when l = 28%. After the second step, about 59.53% of synthetic datasets were finally selected as relevant and the remaining ones were discarded. As for k-NN, in turn, the observed meta-learning accuracy was higher than using all synthetic data for intermediate values of noise level (from 8% to 21%). The best result was 60.57% accuracy when l = 13%. Thus, about 34% of synthetic datasets were discarded and the remaining ones were selected as relevant. The RF meta-learner indeed was less sensitive to the training set of meta-examples. Table II summarizes the accuracy obtained by the k-NN and Random Forest meta-learners when the PU Learning technique was adopted for selecting synthetic datasets in our experiments.

In practice, it is hard to know in advance the noise level to be adopted in the proposed method. It will depend on the expected relevance of the available datasets, or in other words, up to what extent one can trust on the relevance and representativeness of datasets available in repositories like UCI. Results in Fig. 6 suggest, for instance, that about 15% of the real datasets adopted in our experiments are not entirely relevant. Some noise levels (e.g., 50%) seem to be too excessive and were only considered in the experiments in order to evaluate the robustness of the proposed approach. How to estimate the noise level is an open question that will be better investigated in future work.

2) NNDD: Fig. 8 and Fig. 9 present the accuracy obtained by the k-NN and RF meta-learners when the NNDD technique was adopted for selecting synthetic datasets in our experiments. The results revealed, in general, gain in accuracy when the NNDD learning method was compared with the two baselines adopted. A positive impact was observed in meta-learning performance. As in the PU selection, RF obtained the best results when compared to the results obtained by k-NN meta-learner. RF meta-learner appears to be more robust to noisy and redundant instances in the training set, which can be justified by the fact that the RF uses bagging to produce the ensemble components, thus presenting greater robustness regarding the quality of the training sets. After the inclusion of 293 selected synthetic meta-examples by NNDD the RF obtained the accuracy rate of 63.56%, while the k-NN meta-learner obtained an accuracy rate of 61.23%. Finally, better results were obtained by RF when compared to the k-NN. Table III summarizes the accuracy obtained by the k-NN and RF meta-learners when the NNDD method was adopted for selecting synthetic datasets.

Now, we would like to define the superiority of the proposed approach for selecting synthetic datasets. For this, we applied a statistic test. In this paper, we applied the Wilcoxon Rank Sum method, a non-parametrical test which aims to compare two independent samples of the same size or uneven. As a result of the Wilcoxon Rank Sum Test (95% of confidence), it is observed that, in a general way, the new synthetic dataset selection approach is better when compared to the results observed when using all synthetic datasets without selection, from a statistical point of view, presenting p-value < 0.05.

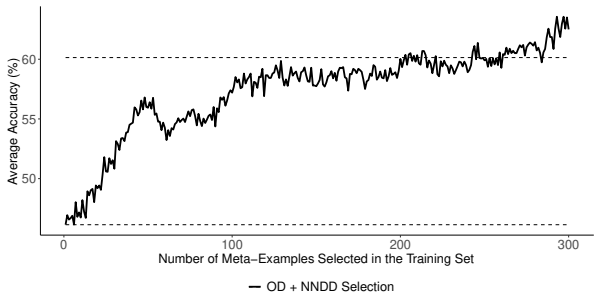

Fig. 8. Average accuracy obtained by Random Forest using NNDD for selecting synthetic datasets. The training set is initialized with all the original datasets, leaving only synthetic datasets for selection. Dashed horizontal lines represent the accuracy obtained using all original datasets and synthetic datasets (top) or just the original datasets (bottom).

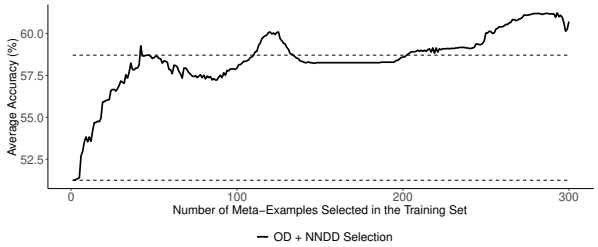

Fig. 9. Average accuracy obtained by k-NN using NNDD to select synthetic datasets. The training set is initialized with all the original datasets, leaving only synthetic datasets for selection. Dashed horizontal lines represent the accuracy obtained using all original datasets and synthetic datasets (top) or just the original datasets (bottom).

VI. Conclusions

In this paper, we presented a new approach for selecting synthetic datasets in meta-learning by combining data manipulation and OCC techniques. The proposal aims to reduce the computational cost of performing experiments and improve the performance of meta-learning by enhancing the number of relevant datasets for generating meta-examples.

We performed experiments in a case study, in which PU learning and the NNDD method were adopted to select the most relevant datasets. The obtained results showed that it is feasible to take advantage of a large number of meta-examples provided by synthetic datasets without having significant additional computational costs. In addition, the performance gain was achieved due to the elimination of irrelevant meta-examples. Improvement in meta-learning performance was observed using the one-class techniques compared with adopted baselines (see Section V-A). The k-NN and Random Forest meta-learners showed a performance gain compared to the results obtained in [2], [23] by using the k-NN algorithm as meta-learner.

TABLE II

ACCURACY RATE (%) OBTAINED BY K-NN AND RANDOM FOREST META-LEARNERS AND THE AVERAGE NUMBER OF EXAMPLES IN THE TRAINING SETS, USING PU LEARNING.

Noise Level (1%)	**k-NN**			**Random Forest**		
/Training data	***OD***	***OD + all***	***OD + PU selection***	***OD***	***OD + all***	***OD + PU selection***
13%	51.25	58.71	**60.57 (+1.86)**	46.20	60.14	58.71 (-1.43)
28%			56.14 (-2.57)			**63.14 (+3)**
Average number of examples	57	933.35	**635.87**	57	933.35	635.87
			578.68			**578.68**

TABLE III

AVERAGE ACCURACY (%) OBTAINED BY RANDOM FOREST AND K-NN META-LEARNERS AND THE AVERAGE NUMBER OF EXAMPLES IN THE TRAINING SET CONSISTING OF NNDD SELECTION AND ORIGINAL DATASETS.

Algorithms/Training data	**OD**	**OD + all**	**OD + NNDD selection**
RF	46.20	60.14 (+13.94)	**63.56 (+3.42)**
k-NN	51.25	58.71 (+7.46)	**61.23 (+2.52)**
Average number of examples	57	933.35	**350**

Finally, we would like to highlight some aspects of our work that will be investigated in the future, such as: (1) initially, we adopted RF and k-NN as meta-learners; however, it is possible that other classifiers with useful potential could be utilised, such as SVMs, for instance; (2) other one-class classification techniques could be investigated in case of different classifiers are adopted as meta-learners; (3) finally, other case studies can be developed in future works.

REFERENCES

[1] P. Brazdil, C. Giraud-Carrier, C. Soares, and R. Vilalta, *(Eds.), Metalearning - Applications to Data Mining*. Springer, Berlin, Heidelberg, 2009.

[2] A. Sousa, R. Prudêncio, and C. S. T. Ludermir, "Active learning and data manipulation techniques for generating training examples in meta-learning," *Neurocomputing, 194.*, pp. p. 45–55, 2016.

[3] S. S. Khan and M. G. Madden, "One-class classification: taxonomy of study and review of techniques," *The Knowledge Engineering Review*, vol. 29, no. 3, p. p. 345–374, 2014.

[4] C. Soares, "Uci++, improved support for algorithm selection using datasetoids." Lecture Notes in Computer Science, 2009, vol. 5476, pp. 499–506.

[5] B. Liu, Y. Dai, X. Li, W. S. Lee, and P. S. Yu, "Building text classifiers using positive and unlabeled examples," *Third IEEE International Conference on Data Mining*, pp. 179–186, 2003.

[6] D. Tax, "One-class classification." *Ph.D. Thesis, Delf University of Technology.*, 2001. [Online]. Available: http://homepage.tudelft.nl/n9d04/thesis.pdf

[7] C. Giraud-Carrier, "Metalearning - a tutorial." *In: Procs. of The Seventh International Conference on Machine Learning and Applications (ICMLA), San Diego, California,USA.*, 2008.

[8] R. Prudêncio and T. Ludermir, "Meta-learning approaches to selecting time series models," *Neurocomputing 61,*, p. p. 121–137, 2004.

[9] R. Caiuta, A. Pozo, L. Emmendorfer, and S. R. Vergilio, "Selecting software reliability models with a neural network meta classifier," *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence.*, 2008.

[10] A. C. A. Nascimento, R. B. C. Prudêncio, I. G. Costa, and M. C. P. de Souto, "Mining rules for the automatic selection process of clustering methods applied to cancer gene expression," *Lecture Notes in Computer Science.*, vol. 5769, pp. 20–29, 2009.

[11] R. R. Parente, A. M. P. Canuto, and J. C. Xavier-Junior, "Characterization measures of ensemble systems using a meta-learning approach," *The 2013 International Joint Conference on Neural Networks (IJCNN), Dallas, Texas, USA*, 2013.

[12] M. Lichman, "UCI machine learning repository (2013)." 2013. [Online]. Available: http://archive.ics.uci.edu/ml

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

[14] K. Hempstalk, E. Frank, and I. H. Witten, "One-class classification by combining density and class probability estimation," in *Machine Learning and Knowledge Discovery in Databases*, W. Daelemans, B. Goethals, and K. Morik, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 505–519.

[15] B. Liu, W. S. Lee, P. S. Yu, and X. Li, "Partially supervised classification of text documents," *ICML '02 Proceedings of the Nineteenth International Conference on Machine Learning*, pp. p. 387–394, 2002.

[16] P. Perera and V. M. Patel, "Learning deep features for one-class classification," *IEEE Transactions on Image Processing*, vol. 28, no. 11, pp. 5450–5463, 2019.

[17] Q. Wei, Y. Ren, R. Hou, B. Shi, J. Y. Lo, and L. Carin, "Anomaly detection for medical images based on a one-class classification," *Medical Imaging 2018: Computer-Aided Diagnosis, vol. 10575, International Society for Optics and Photonics*, 2018.

[18] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson, "Estimating the support of a high-dimensional distribution," *Neural Computation*, vol. 13, no. 7, pp. 1443–1471, 2001.

[19] K. Crammer and G. Chechik, "A needle in a haystack: Local one-class optimization," *ICML '04 Proceedings of the twenty-first international conference on Machine learning*, p. p. 26 , 2004.

[20] M. Yousef, S. Jung, L. C. Showe, and M. K. Showe, "Learning from positive examples when the negative class is undetermined- microrna gene identification," *Algorithms for Molecular Biology*, vol. 3, no. 2, 2008.

[21] R. Xia, X. Hu, J. Lu, J. Yang, and C. Zong, "Instance selection and instance weighting for cross-domain sentiment classification via pu learning," *Proceedings of the Twenty-Third international joint conference on Artificial Intelligence (IJCAI)*, pp. 2176–2182, 2013.

[22] C. Gong, T. Liu, J. Yang, and D. Tao, "Large-margin label-calibrated support vector machines for positive and unlabeled learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 11, pp. 3471–3483, 2019.

[23] R. Prudêncio, C. Soares, and T. Ludermir, "Uncertainty sampling methods for selecting datasets in active meta-learning," *in: Proceedings of the 2011 International Joint Conference on Neural Networks*, p. 1082–1089, 2011.

[24] P. Brazdil, C. Soares, and J. P. da Costa, "Ranking learning algorithms: Using ibl and meta-learning on accuracy and time results," *Machine Learning*, vol. 50(3), pp. 251–277, 2003.

[25] I. H. Witten, E. Frank, M. A. Hall, and C. J. Pal, *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, 2017.

[26] T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE Transactions on Information Theory*, no. 1, pp. 21–27, 1967.