Pattern Recognition Letters 125 (2019) 813–820

Contents lists available at ScienceDirect

Pattern Recognition Letters

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

Reverse-nearest neighborhood based oversampling for imbalanced, multi-label datasets

Payel Sadhukhan[a,*], Sarbani Palit[b]

[a] Machine Intelligence Unit, Indian Statistical Institute, Kolkata, India
[b] Computer Vision and Pattern Recognition Unit, Indian Statistical Institute, Kolkata, India

ARTICLE INFO

Article history:
Received 20 February 2019
Revised 25 July 2019
Accepted 6 August 2019
Available online 6 August 2019

Keywords:
Reverse nearest neighborhood
Multi-label classification
Multi-label learning
Class-imbalance
Oversampling

ABSTRACT

In this article, we present a novel reverse-nearest neighborhood based oversampling scheme for the imbalanced labels of a multi-label dataset. Reverse nearest neighborhood of a query point includes all those points which contain the query point as one of their neighbor. It facilitates us to identify an adaptive number of neighbors (according to the density and distribution of points) instead of a fixed number of neighbors. We add label-specific synthetic minority instances in the reverse nearest neighborhood of the minority points of each label. Reverse nearest neighbor configuration also detects the singular minority points, which we avoid as seed points in the oversampling phase. On the oversampled data of each label, we train and invoke a Linear Support Vector Machine to complete the learning and testing. Results of the proposed method against comparing methods on class-imbalance focused metrics indicates its competence in handling differently imbalanced multi-label datasets.

© 2019 Elsevier B.V. All rights reserved.

1. Introduction

Multi-label datasets have gained the attention of the machine learning community in the past two decades. A multi-label dataset is characterized be the membership of an instance to more than one overlapping labels. Let $\mathcal{D}$ be a multi-label dataset, $\mathcal{D} = \{(\mathbf{x}_i, \mathcal{Y}_i) : i = 1, 2, \ldots, n\}$, where $\mathbf{x}_i$ denotes the ith instance and $\mathcal{Y}_i$ represents its label vector. If a dataset $\mathcal{D}$ has $\mathcal{L}$ labels, $\mathcal{Y} = \{y_{i1}, y_{i2}, \ldots, y_{iL}\}$, y_{ij} denotes the jth label membership of instance $\mathbf{x}_i$. We have considered two class multi-label problems, each label can take exactly one of two values, 0 (negative) or 1 (positive). Inception of multi-label research was started with text categorization [1]. Other real-world multi-label sources include music [2], image [3,4], bioinformatics [5] and medical [6].

For effective handling of multi-label datasets, we have to analyze the intrinsic characteristics of the multi-label datasets. A prevalent characteristic of multi-label datasets is class imbalance of the labels. In imbalanced datasets, quantitative disproportion exists in the distribution of different classes. If we consider a two-class dataset, the quantitatively abundant class and the quantitatively scarce class are termed as majority class and minority class respectively. Let *imb* be the imbalance ratio a dataset. It is the ratio of number of instances in the majority class of a dataset to the number of instances in it's minority class. In regular datasets we have exactly one label, hence we have a single imbalance ratio. In a multi-label dataset, class-membership of the instances vary from label to label. Different are labels deal with differing aspects associated with a dataset. This leads to different positive and negative class cardinalities for different labels, and consequently different imbalance ratios are obtained across the labels. Generally speaking, for most of the labels of the multi-label datasets, the positive class remains under-represented and becomes the minority class. But at times, in a few cases, the negative class also becomes the minority class. The differential imbalance ratios coupled with the above stated class-switching for different labels make the class imbalance problem more critical as well as interesting. In recent years, a few works like COCOA [7] and [8] have given attention to this pertinent aspect.

Oversampling the minority class has been a popular technique for handling imbalance of datasets and learning the minority instances. Starting with SMOTE [9], several and diversified oversampling techniques have developed over the years to address the issue of imbalance of datasets. In oversampling, synthetic points are added to increase the cardinality as well as representation of the under-represented minority class. In this paper, we introduce an imbalance-adaptive oversampling scheme for multi-label datasets. We oversample a set of label-specific synthetic minority points, one for each label and use them (along with original training set) to train a set of first-order classifiers. The novelty of our scheme lies with the use of reverse-nearest neighborhood exploration.

* Corresponding author.
E-mail address: payel_r@isical.ac.in (P. Sadhukhan).

https://doi.org/10.1016/j.patrec.2019.08.009
0167-8655/© 2019 Elsevier B.V. All rights reserved.

Reverse k-nearest neighbor of an instance **p** are all those instances which have **p** as one of their k-nearest neighbors. The number of RkNNs of **p** is not necessarily k and its value is auto-regulated by the local density and distribution of points around **p**. Reverse nearest neighborhood principles are used to demarcate the neighborhood as well as neighbors of a candidate minority point, centering which the oversampling will be performed. In regular SMOTE and SMOTE-inspired oversampling techniques, a generic 'k' value or neighborhood size is selected across the entire dataset. Reverse nearest neighborhood allows us an adaptive neighborhood size for performing the oversampling. For each label, the synthetic minority points are oversampled from the reverse-nearest neighborhood of the minority instances. To cope with the differential imbalance problem of multi-label datasets, we oversample a dedicated set of synthetic instances for each label. For each label we train a classifier using the original set of instances and the label specific synthetic minority set. We have presented and compared our results with several state-of-the-art multi-label learners as well as class-imbalance handling learners. Our method has delivered statistically better performance with respect to all the compared learners on Macro F_1 metric. On Macro AUC, the proposed method has delivered comparable to statistically significant improvement. In the next section, we discuss the extant works in the domain of multi-label learning as well as reverse nearest neighborhood. In Section 3 we discuss the fundamentals of reverse nearest neighborhood in brief. The proposed scheme is presented in Section 4 followed by Experimental Study in Section 5. Section 6 presents the results and analysis. The article is wrapped by the Conclusion in Section 7.

2. Related works

Since their inception in the beginning of last century, multi-label datasets have been tried with several and diversified techniques for fruitful learning. A detailed analysis of the techniques can be found in [10] and we provide a brief review of the extant works in the next paragraph. The scheme proposed in this work uses a relatively new technique, reverse nearest neighborhood of data points. We have also provided a brief review of the works that have used reverse nearest neighborhood scheme to solve some machine learning subtask in the later part of this section.

Multi-label learning:

According to the literature, multi-label learners are principally classified into two types- *Problem transformation (PT) approach* and *Algorithm Adaptation (AA) approach*. In *problem transformation* approach, a multi-label dataset is re-structured and decomposed for learning using a regular classification algorithm. Problem transformation approaches can be further classified into *-first-order, second-order* and *higher-order* according to the number of labels involved in each classifier. In first-order decomposition, we train $\mathcal{L}$ classifiers for $\mathcal{L}$ labels of a dataset. Binary Relevance (B_R) [11] is an example of this type. The principal drawback of first-order problem transformation approach is their inability to capture label correlation. Many advanced multi-label algorithms like feature extraction and selection are designed in this framework to analyze their intrinsic capabilities on learning multi-label datasets [12]. In second-order decomposition, the labels are considered in pair-wise fashion and we train one classifier for each label-pair. Such techniques capitalizes the pair-wise correlation of the labels. Examples are Calibrated Label Ranking [13] and Efficient Pairwise Classification [14]. In higher order approach, more than two labels are learned together to facilitate the correlations of a bigger subset of labels. Rakel [15], Classifier chains [16,17], Instance based logistic regression (IBLR) [18] and Ensemble learners are other pertinent examples of this type. In *algorithm adaptation approach*, an existing machine learning technique is modified in the context of multi-label learning. Notable examples are MLkNN [19], BPMLL [20] and Naive–Bayes [21].

Apart from the above described paradigms, feature selection and extraction have become popular as tools for effective learning of multi-label datasets. In these algorithms, instead of using a single feature set for all labels a dedicated feature sets are selected or extracted for different labels to improve the learning. Notable examples include [22–24]. A survey of the extant works in multi-label feature extraction and selection in first-order framework can be found in [12]. Joint feature selection and classification (JFSC) [25,26] are two recent works which perform label-correlated feature selection of multi-label datasets. Multi-label classification is a pertinent characteristic for images and has gained sufficient interest at recent times. [27] has proposed an image dataset which has a generic multi-label framework. Two recent works [3] and [4] have used neural networks to learn the various components of an image and subsequently perform multi-label classification of the image objects. Class distribution of the various labels of a multi-label dataset is an varying and interesting parameter. As said earlier, multi-label datasets are class-imbalanced and they are differently imbalanced also. COCOA [7] facilitated multi-label learning by working on the imbalance issue and considering an ensemble of classifiers using pair-wise label correlation. Couple of more works, Wu and Lin [28] and Huang and Lin [29] have used cost-adaptive paradigm to address multi-label problems. Methods involving techniques like convex relaxation [30] and ensembles of random graph [31] are also used for addressing multi-label classification.

Reverse nearest neighbor principles: Technically, reverse nearest neighborhood might just seem a flip side of the k-nearest neighborhood, but it gives a considerable amount of information perceptually (which is not available from the data of k-nearest neighborhood). Outlier detection in an unsupervised context and in data streams is implemented using reverse-nearest neighborhood by [32] and [33] respectively[34]. has worked towards efficient reverse nearest search in metric spaces. Reverse nearest neighbor principles has been used in biological context of protein information mining by Ning et al. [35]. Problems on spatial data search is also addressed by the same in [36]. Reverse-nearest neighbor based algorithms have solved spatio-temporal query and range queries in [37]. Bryant and Cios [38] has implemented DB-SCAN algorithm via RkNN. In [39], the authors have used locality-sensitive hashing (LSH) to find the reverse k-nearest neighbors of a query point. Hashing is a popular technique for quick retrieval of the k-nearest neighbors in a big data context. In reverse k-nearest neighbor principles the neighborhood size is adaptive to the location of the query point. An efficient hashing technique incorporating query-adaptive and fine-grained ranking information are designed by Liu et al. [40]. Another pertinent work in hashing context is [41]. Liu et al. [41] provides a distributed framework for hashing and neighbor identification in large-scale datasets.

3. Fundamentals of Reverse-nearest neighborhood

Definition 1. : Given a set of instances $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$ ($\mathcal{X} \subset \mathbb{R}^N$) and a point **p** ($\mathbf{p} \in \mathbb{R}^N$), a Reverse Nearest Neighbor query concerning **p** in search space $\mathcal{X}$ retrieves all the points $\mathbf{x}_i \in \mathcal{X}$ that have **p** as their nearest neighbor. Thus, a Reverse k-Nearest Neighbor (RkNN) search returns all those points $\mathbf{x}_i \in \mathcal{X} (i = 1, 2, \ldots, n)$ whose k-nearest neighborhood contain **p**.

In k-nearest neighborhood of a point, exactly k points can be present in its neighborhood. These k points are closest to **p** than other points in the search space. On the contrary, when the cardinality of the search space is n, reverse k-nearest neighborhood of **p** can contain 0 to n points. RkNN dose not define any fixed number of neighbors around query point **p**. The number of neighbors

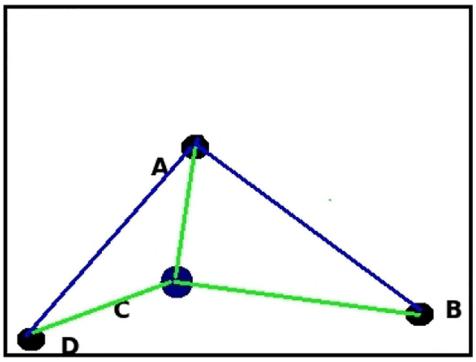


Fig. 1. We want to find the RkNNs of point C. We consider k=1. It can be found visually that the Euclidean distance between A and C is less than between B and C as well as D and C. If we were to find 1-nearest neighbor of C, A would have been the answer. Now, let us analyse with respect to RkNN. Though A is closet to C but C is not closest to A, A has some nearer points or neighbors in its vicinity. Hence, A is not a RkNN of C. Though B may seem far off from C, it becomes a RkNN of C because C is nearer to B than any other given point in the feature space. The same holds for D also, hence D becomes a RkNN of C. At k value 1, C has 2 RkNNs {B, D}. RkNN gives a density based neighborhood estimation of the points.

around a point varies from location to location and it depends on the relative configuration of the query point and the points in the search space. The R*k*NN count of **p** is adaptive to the density and distribution of points around **p** and gives a more versatile estimation of neighborhood than *k*-nearest neighborhood based scheme. An example is illustrated in Fig. 1.

4. Proposed method

Motivated to handle the differential class imbalance of multi-label learning, we present an oversampling scheme which generates a dedicated set of synthetic minority instances for each label. A dedicated set of synthetic minority instances for each label is more effectual than a single synthetic minority set for all labels. We can generate the minority set according to the requirement of a label. It allows us to set the size of the synthetic minority set of a label in consonance with the imbalance ratio of that label. Let us have two labels a and b with different imbalance ratios *imb1* and *imb2* respectively (where $imb1 < imb2$). We need to add more points in the synthetic minority set of b than that of a in order to equate the cardinalities of their respective majority and minority classes. We assume that the positive class is the minority class across all labels. The number of synthetic minority points we generate is dependent on the difference in cardinality between the majority (negative) and minority (positive) sets. If the number of points in the positive class is more than or equal to the number of points in the negative class for a label, we consider the imbalance ratio to be 1 and do not add any synthetic minority points for that label. Our scheme is somewhat motivated by SMOTE, but the neighborhood exploration of our scheme is entirely different from SMOTE [42]. SMOTE relies on *k*-nearest neighborhood of points whereas our scheme uses a more versatile reverse-nearest neighborhood principles to identify the neighborhood of points. (We have included SMOTE in the comparative study. RkNN principles accommodates a varying number of neighbors for different points in a dataset. The RkNN count of a point depends on its proximity or presence in the neighborhood of remaining points in the search space. The significant improvement delivered by RkNN-based neighborhood exploration over k-NN based SMOTE is manifested there.)

Here, we describe the scheme for one label, say *j*. Let $\mathcal{D}$ be the training set. For $\mathcal{L}$ labels, the procedure is repeated $\mathcal{L}$ times to generate $\mathcal{L}$ sets of synthetic minority points, one for each label. Let $\mathbf{Majority}_j$ and $\mathbf{Minority}_j$ be the sets of majority and minority instances of label *j* respectively.

$$\mathcal{D} = \mathbf{Majority}_j \cup \mathbf{Minority}_j \tag{1}$$

Let N_j be the cardinality difference of $\mathbf{Majority}_j$ and $\mathbf{Minority}_j$.

$$N_j = |\mathbf{Majority}_j| - |\mathbf{Minority}_j| \tag{2}$$

To equate the cardinalities of the majority and minority sets, we will add N_j points in the synthetic minority set of label *j*. Here, $\mathbf{S}_j$ denotes the synthetic minority set of label *j*.

- **Reverse k-nearest neighbors of original minority points**: For each (minority) point $\mathbf{x}_i$ in $\mathbf{Minority}_j$, we will find its R*k*NN from the same set. Let $\mathbf{R}_j(\mathbf{x}_i)$ and $RC_j(\mathbf{x}_i)$ denote the set of reverse k-nearest neighbors of $\mathbf{x}_i$ and its cardinality respectively.

$$\mathbf{R}_j(x_i) = \{\mathbf{x}_k, \mathbf{x}_k \text{ is a RkNN of } \mathbf{x}_i, \ \ \mathbf{x}_k \in \mathbf{Minority}_j\} \tag{3}$$

$$RC_j(x_i) = |\mathbf{R}_j(\mathbf{x}_i)| \tag{4}$$

Since we are querying the minority points in context of $\mathbf{Minority}_j$, the R*k*NN count of each query point will be atleast 1. We will mark the points whose R*k*NN count is exactly 1. Such points are isolated from the other points in the search space and do not have any R*k*NN other than itself. Let the set of such points for label *j* be $\mathbf{M}_j$.

$$\mathbf{M}_j = \{\mathbf{x}_i, \mathbf{x}_i \in \mathbf{Minority}_j \text{ and } RC_j(\mathbf{x}_i) = 1\} \tag{5}$$

As each member of $\mathbf{M}_j$ lies all by itself (without any minority neighbors in the vicinity) in the feature space, they do not serve as good candidates for adding the synthetic minority points and we do not consider them as seed points for adding the synthetic minority points. Let $\mathbf{Aug_Minority}_j$ be the set of points around which we will perform the oversampling. We compute $\mathbf{Aug_Minority}_j$ by removing the elements of $\mathbf{M}_j$ from $\mathbf{Minority}_j$.

$$\mathbf{Aug_Minority}_j = \mathbf{Minority}_j \setminus \mathbf{M}_j \tag{6}$$

- **Generating synthetic minority points**: To generate a synthetic minority point, we random select a point from $\mathbf{Aug_Minority}_j$, **q** (say) and find its R*k*NNs. We randomly select one of the R*k*NN of **q**. Let that point be **r**. We must maintain $\mathbf{q} \neq \mathbf{r}$, if so we will select repeat a original minority point **q** in the synthetic minority set. Let d(**q, r**) be the euclidean distance between **q** and **r**. We generate a random value ν such that $0 < \nu < \text{d}(\mathbf{q}, \mathbf{r})$. The synthetic point, **new** is generated at a random distance ν from **q** on the direction vector of **q** and **r**.

$$\mathbf{new} = \mathbf{q} + (\mathbf{r} - \mathbf{q}) * \nu \tag{7}$$

We generate N_j number of such points to form $\mathbf{S}_j$, the synthetic minority set of label *j*. While selecting each synthetic point, we sequentially consider each point of $\mathbf{Aug_Minority}_j$. This is done to maintain the uniformity of locations of the synthetic minority points. The same procedure is repeated across the entire label set to select $\mathcal{L}$ sets of synthetic minority points $\mathbf{S}_1, \ \mathbf{S}_2, \ldots, \mathbf{S}_{\mathcal{L}}$.

- **Classifier modeling**: We run the classifiers in first-order setting, one classifier for each label. For each label *j*, we get the oversampled instance set, $\mathbf{O}_j$ taking the union of original instance sets $\mathbf{Majority}_j$ and $\mathbf{Minority}_j$ along with the synthetic minority set $\mathbf{S}_j$.

$$\mathbf{O}_j = \mathbf{Majority}_j \cup \mathbf{Minority}_j \cup \mathbf{S}_j \tag{8}$$

We train a Linear SVM classifier for label j using $\mathbf{O}_j$ to get the classifier model M_j. The label predictions of test set corresponding to label *j* are obtained by invoking M_j. The procedure is repeated over L labels.

Table 1
Description of Datasets for experiment on imbalance. *D, L* and *F* denote the number of instances, labels and features respectively. *L.Card* and *L.Uniq* gives the values of average number of instances per label and the number of unique label combinations respectively. min IR, max IR and avg IR denotes the minimum, maximum and average imbalance ratio of a dataset respectively.

Dataset	Domain	Att.type	(*D*)	(*L*)	(*F*)	(*L.Card*)	(*L.Uniq*)	min IR	max IR	avg IR
CAL500	music	numeric	502	124	68	25.058	502	1.04	24.39	3.85
Corel5k	image	nominal	5000	44	499	3.522	3175	3.46	50	17.86
Emotions	music	numeric	592	6	72	1.869	27	1.25	3.0	2.15
Enron	text	nominal	1702	24	1001	3.378	753	1.0	43.48	5.35
medical	bio-NLP	nominal	978	14	1449	1.075	42	2.67	43.48	11.24
RCV1 Subset1	text	numeric	6000	472	43	2.458	574	3.34	50.0	15.15
RCV1 Subset2	text	numeric	6000	472	39	2.170	489	3.22	47.62	15.87
Slashdot	text	nominal	3782	22	1079	1.134	118	5.46	35.71	10.99
Tmc2007	text	nominal	28596	500	15	2.100	637	1.447	34.483	5.848
Yeast	biology	numeric	2417	13	103	4.233	189	1.32	12.5	2.78

Table 2
Predictive performance of methods in terms of multi-label evaluation metrics – ↑ indicates higher is better and ↓ indicates lower is better, best outcome is indicated in bold-face.

	Corel5k	Enron	Medical	Slashdot	Tmc2007
Macro F_1 ↑					
LIFT	0.117 0.006	0.300 0.009	0.577 0.008	0.378 0.010	0.574 0.006
CLR	0.058 0.007	0.227 0.003	0.652 0.008	0.233 0.005	0.618 0.007
RAKEL	0.088 0.007	0.248 0.005	0.577 0.015	0.246 0.004	0.642 0.004
IBLR	0.061 0.005	0.218 0.006	0.562 0.008	0.278 0.006	0.501 0.005
COCOA	0.196 0.003	0.326 0.005	0.689 0.004	0.325 0.003	**0.671 0.003**
SMOTE	0.127 0.003	0.269 0.005	0.686 0.005	0.327 0.009	0.576 0.007
ADASYN	0.198 0.005	0.338 0.005	0.752 0.011	0.373 0.008	0.564 0.005
USAM	0.142 0.002	0.269 0.004	0.670 0.005	0.256 0.007	0.341 0.006
Proposed method	**0.201 0.003**	**0.345 0.006**	**0.768 0.007**	**0.442** 0.006	0.631 0.005
Macro AUC ↑					
	Mean std	Mean std	Mean std	Mean std	Mean std
LIFT	0.708 0.007	0.698 0.009	0.950 0.014	0.832 0.009	0.904 0.007
CLR	**0.738 0.003**	0.707 0.004	0.942 0.009	0.676 0.005	0.907 0.007
RAKEL	0.552 0.006	0.643 0.006	0.826 0.011	0.614 0.004	0.902 0.006
IBLR	0.651 0.006	0.686 0.005	0.924 0.004	0.623 0.004	0.876 0.009
COCOA	0.718 0.004	0.730 0.003	0.957 0.005	0.732 0.004	**0.921 0.004**
SMOTE	0.600 0.004	0.616 0.006	0.874 0.012	0.692 0.005	0.788 0.005
ADASYN	0.702 0.006	0.721 0.003	0.953 0.008	0.788 0.005	0.861 0.007
USAM	0.576 0.004	0.602 0.011	0.857 0.012	0.614 0.007	0.802 0.004
Proposed method	0.722 0.003	**0.735 0.006**	**0.977 0.005**	**0.820 0.005**	0.901 0.004

5. Experimental study

5.1. Datasets

We have used 10 real-world multi-label datasets in the empirical study. The feature sets of these datasets are of purely nominal or purely numeric nature. To maintain uniformity, we have pre-processed these datasets as per the recommendations in class-imbalanced focused multi-label work COCOA [7]. The labels which have less than 20 instances in the minority class or possess imbalance ratio higher than 50 are removed. Feature sets are also reduced for large datasets namely RCV1 Subset1, RCV1 Subset2 and Tmc. The detailed statistics of 10 datasets are presented in Table 1. These datasets are obtained from MULAN [43] and MEKA [44] repositories.

5.2. Comparing methods

We have included five state-of-the-art multi-label learners in the experiments. Besides these, we have modified two over-sampling based class-imbalance learners, SMOTE and ADASYN in multi-label context. We have also included undersampling based class-imbalance learner, USAM.

Multi-Label Learning with Label-Specific Features (LIFT) [22]: This method extracts the positive and negative cluster centers for each label and uses them to compute the label-specific features of each label. These label-specific features are used to train the classifier set. Test instances are subsequently mapped in a similar fashion for each label. Classifier is invoked for each label to get the prediction. As per the recommendations in the paper, we have used $r = 0.1$.

Random k-Labelsets: An Ensemble Method for Multi-label Classification (RAKEL) [15]: This is a higher-order method which considers a number of overlapping subsets of labels. Each subset is learned and the cumulative predictions of all the subsets are used to find the final prediction. For L labels, we set $k = 3$ and ensemble size to $2 \times L$ per the authors' recommendations.

IBLR [18]: It is a higher-order approach which combines model-based and similarity-based inference to facilitate multi-label classification.

Towards Class-Imbalance Aware Multi-Label Learning (COCOA) [7]: This is one of the introductory work in class-imbalance focused multi-label learning. It takes into account both label-correlation and imbalance ratio of a label to make an imbalance ratio of a label to make an imbalance-aware prediction. We have used min(L-1,10) coupling class labels and 10 models per ensemble for all datasets.

Table 3
Predictive performance of methods in terms of multi-label evaluation metrics – ↑ indicates higher is better and ↓ indicates lower is better, best outcome is indicated in bold-face.

	CAL500	Emotions	RCV1 Subset1	RCV1 Subset2	Yeast
Macro F_1 ↑					
	Mean std	Mean std	Mean std	Mean std	Mean std
LIFT	0.075 0.003	0.631 0.007	0.201 0.011	0.198 0.009	0.391 0.006
CLR	0.080 0.004	0.598 0.007	0.226 0.005	0.226 0.007	0.415 0.005
RAKEL	0.194 0.008	0.616 0.016	0.274 0.006	0.267 0.004	0.418 0.004
IBLR	0.221 0.006	0.621 0.007	0.198 0.006	0.195 0.003	0.409 0.003
COCOA	0.208 0.007	0.662 0.009	0.364 0.005	0.342 0.005	0.457 0.008
SMOTE	0.231 0.006	0.582 0.006	0.313 0.006	0.308 0.004	0.431 0.006
ADASYN	0.243 0.006	0.624 0.007	0.352 0.005	0.348 0.006	0.461 0.007
USAM	0.219 0.005	0.589 0.003	0.317 0.006	0.309 0.004	0.431 0.007
Proposed method	**0.246 0.003**	**0.682 0.006**	**0.478 0.006**	**0.475 0.010**	**0.487 0.008**
Macro AUC ↑					
	Mean std	Mean std	Mean std	Mean std	Mean std
LIFT	0.533 0.007	0.834 0.009	0.895 0.008	0.894 0.006	0.678 0.005
CLR	**0.560 0.003**	0.798 0.003	0.890 0.004	0.881 0.006	0.651 0.003
RAKEL	0.531 0.006	0.792 0.010	0.725 0.003	0.722 0.004	0.638 0.004
IBLR	0.509 0.006	0.835 0.008	0.786 0.008	0.792 0.004	0.691 0.007
COCOA	0.554 0.006	0.838 0.008	0.888 0.004	0.886 0.003	**0.712 0.005**
SMOTE	0.514 0.003	0.706 0.016	0.624 0.007	0.620 0.009	0.581 0.007
ADASYN	0.536 0.009	0.801 0.008	0.831 0.007	0.832 0.006	0.651 0.003
USAM	0.516 0.004	0.708 0.015	0.674 0.005	0.672 0.005	0.579 0.004
Proposed method	0.556 0.006	**0.841 0.009**	**0.912 0.008**	**0.914 0.007**	0.658 0.010

Calibrated Label Ranking (CLR) [13]: This is a second-order approach which considers pair-wise correlation of labels. The relevant and non-relevant labels are differentiated via a synthetic bi-partitioning label.

Synthetic Minority Over-sampling Technique (SMOTE) [42]: A state-of-the-art oversampling approach to tackle class-imbalance problems of datasets. We have performed label-specific oversampling of the minority instances depending on the imbalance ratio of labels. We have used neighborhood size 5, $k = 5$. Linear SVM classifier for each label is used to make the label predictions.

Adaptive Synthetic Sampling Approach for Imbalanced Learning (ADASYN) [45]: An effective extension of SMOTE which takes into account the density of minority instances before adding the synthetic points. Similar to SMOTE, we set $k = 5$ and invoke a linear SVM classifier to complete the learning.

Random Undersampling (USAM) [9]: Random under-sampling of the majority class of each label is done to equate the cardinalities of the majority and minority classes. The procedure is repeated across the entire label set to get dedicated undersampled training set for each label. Linear SVM classifier is used for prediction.

Proposed method: Our scheme also requires the user-provided value of neighborhood size. We have set $k = 5$ to maintain a parity with the two other comparing oversamplers, SMOTE and ADASYN. After generating the oversampled training set, we use a linear SVM classifier for learning. In ADASYN, SMOTE and proposed method, number of synthetic minority points generated for a label is equal to the difference in cardinality between its respective majority and minority sets. Similarly, in undersampling method USAM, we have removed majority instances till the cardinality difference between minority and the undersampled majority set is zero.

5.3. Evaluating metrics:

For class-imbalance focused multi-label performance of the above mentioned schemes, we have employed Macro F_1 and Macro AUC. They are described below.

Macro F_1**:** It calculates the average of F_1 values across all labels. Let tp_j, tn_j, fp_j and fn_j denote the number of true positive, true negative, false positive and false negative predictions for label j respectively.

We calculate F_1 for label j,

$$F_{1j} = \frac{2 * tp_j}{2 * tp_j + fp_j + fn_j} \tag{9}$$

$$Macro\ F_1 = \frac{1}{L}\sum_{j=1}^{L} F_{1j} \tag{10}$$

Macro AUC: Let AUC_j be the AUC score for label j. We calculate the average AUC score of all labels in Macro AUC. Higher the value of Macro AUC, better is the performance of the learner.

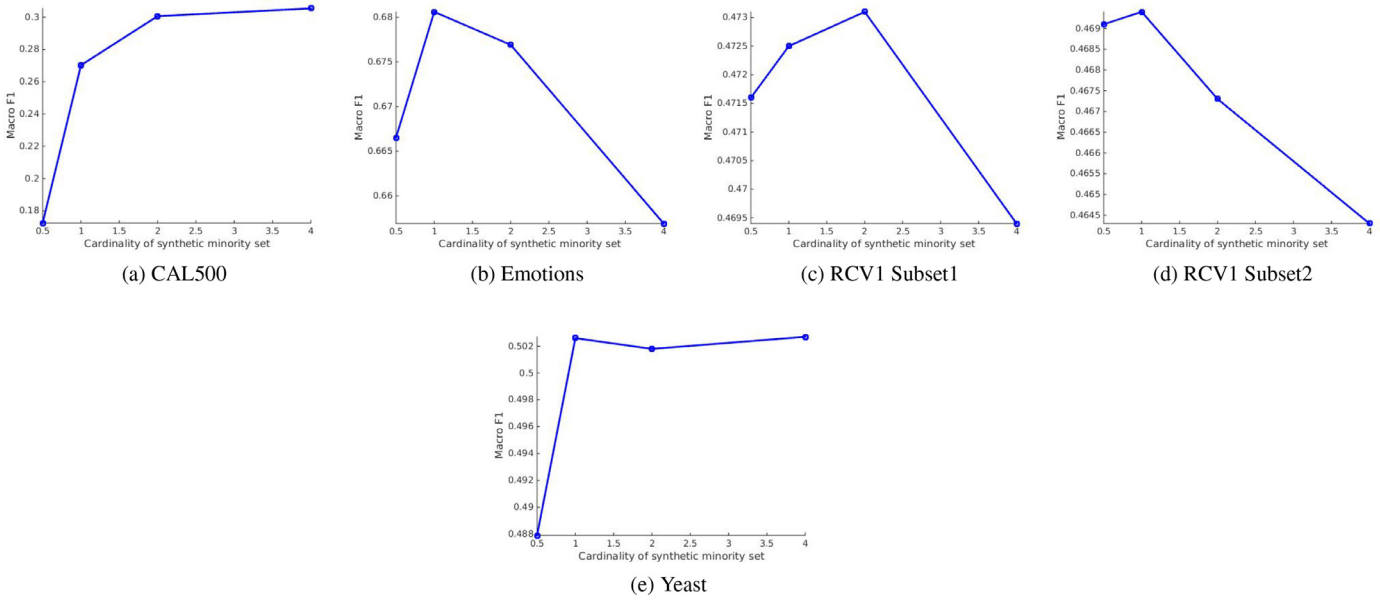
$$Macro\ AUC = \frac{1}{L}\sum_{j=1}^{L} AUC_j \tag{11}$$

Table 4
This table reports the percentage of improvement or percentage of loss of performance given by the proposed method. Percentage of improvement is reported (with respect to second best method) when the proposed method gives best figures for a dataset. When the proposed method fails to give the best values, percentage of loss of performance (with respect to the figures of best performing method) by the proposed method is reported. A positive value denotes improved performance of the proposed method while a negative value signifies loss of performance of the proposed method.

	Macro F_1	Macro AUC
Corel5K	1.50%	−2.17%
Enron	2.07%	0.68%
Medical	2.12%	2.52%
Slashdot	18.50%	1.45%
Tmc	−5.96%	−2.17%
CAL500	1.23%	−0.36%
Emotions	3.02%	0.36%
RCV1 Subset1	31.40%	1.90%
RCV1 Subset2	38.89%	2.24%
yeast	6.56%	−7.58%

Table 5
This table reports the p value at which we can reject the null hypothesis and claim that the performance of the proposed method is statistically different from that of a comparing method. Lower the p value, more significant is the difference. Statistically significant difference with respect to $p = 0.05$ are indicated in boldface.

	LIFT	CLR	RAKEL	IBLR	COCOA	SMOTE	ADASYN	USAM
Macro F_1								
Proposed method	**0.002**	**0.002**	**0.002**	**0.002**	**0.030**	**0.002**	**0.002**	**0.002**
Macro AUC								
Proposed method	0.090	**0.037**	**0.004**	**0.010**	0.201	**0.002**	**0.002**	**0.002**



(a) CAL500 (b) Emotions (c) RCV1 Subset1 (d) RCV1 Subset2

(e) Yeast

Fig. 2. Macro F_1 results of the numeric datasets for various cardinalities of the synthetic minority set. The value in the x-axis is indicative of the cardinality of the $\mathbf{S}_{\min}$. Let *diff* be the difference between minority and majority sets. At x-axis value k, the cardinality of $\mathbf{S}_{\min}$ is $k \times diff$. k or x-axis co-ordinates are varied between 0.5, 1, 2 and 4.

6. Results and analysis

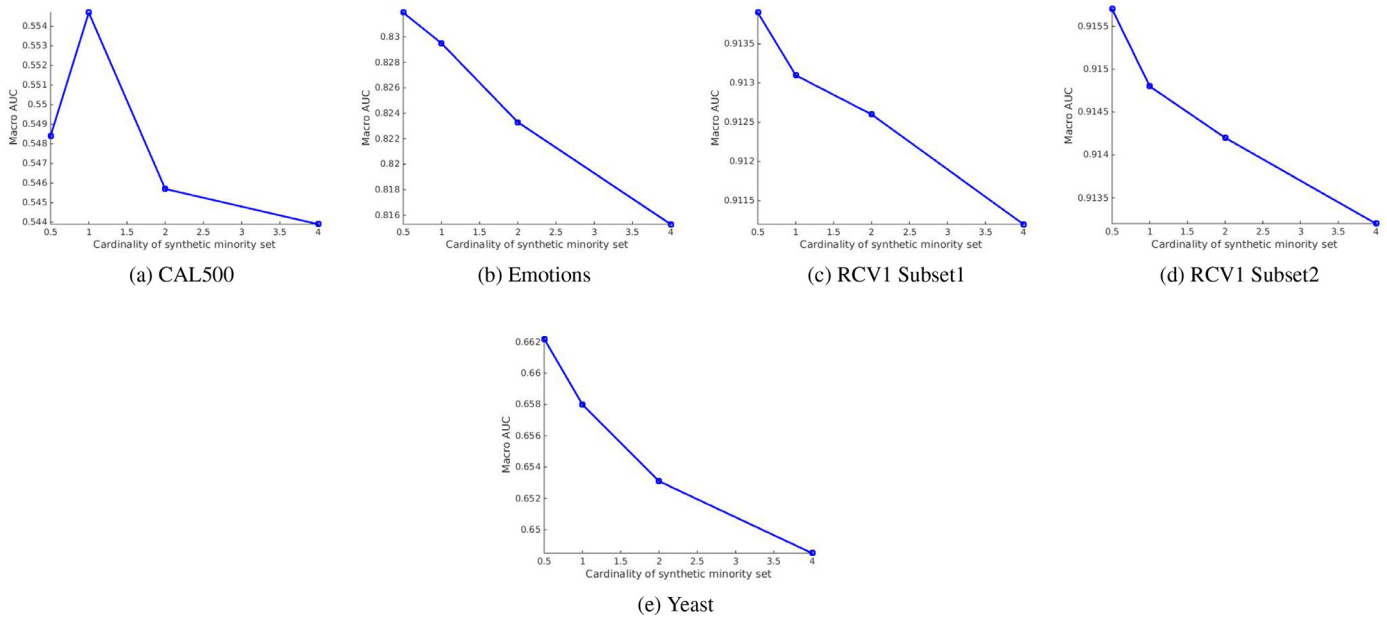
Tables 2 and 3 show the performances of the proposed and comparing methods on 10 datasets. Tables 2 and 3 are dedicated to nominal and numeric datasets respectively. The upper segment of each table is dedicated to Macro F_1 and the lower segment records the Macro AUC scores. For each method and dataset, we have performed 10 runs and have reported the mean and standard deviation values in the rightful location. For nominal datasets with respect to Macro F_1, the proposed method has achieved best scores in 80% (4 out 5) cases. In the remaining one dataset (Tmc), COCOA delivered the best figure. With respect to Macro AUC, the proposed method performed better han others on 60% (3 out of 5) datasets. On the remaining two datasets Corel5k and Tmc, best scores were delivered by CLR and COCOA respectively. From a qualitative point of view, the proposed method has given minor (2.55%, Corel5k, Macro F_1) to significant improvement (36%, Slashdot, Macro F_1) in the cases where it has performed best.

The proposed method has delivered best performances on Macro F_1 metric for all five numeric datasets. Macro AUC scores obtained by the proposed method are better than that of comparing methods in 60% cases (3 out of 5 datasets). COCOA and CLR has achieved the best value of Macro AUC on yeast and CAL500 datasets respectively.

Table 5 reports the outcomes of Wilcoxon Sigmoid Rank Sum Test on pair-wise performances of the proposed method and a comparing method. We have eight comparing methods and two metrics, hence we have conducted sixteen (8×2) pair-wise tests to find the statistical significance of the performance difference between the proposed method and a comparing method, for each metric, Macro F_1 and Macro AUC. Each element of Table 5 shows the p value at which the performance of the two methods are different. Lower the p value, more significance is the difference or more certain we are about rejecting the null hypothesis. The null hypothesis assumes that the performance of two methods are same. p value 0.05 or 5% significance level is the standard threshold for rejecting or accepting a null hypothesis. On Macro F_1, the proposed method's performance is statistically significant and better than that of all methods, obtained p values are less than 0.05 in all cases. On Macro AUC, the proposed method's performances are statistically superior to all but two methods, LIFT and COCOA.

Table 4 gives the quantitative figures about the improvement or loss of performance of the proposed method against the second-best or best method respectively on each dataset and metric pair.

Results on varying the cardinality of synthetic minority set:
We have carried this experiment to study the effect of the cardinality of synthetic minority set on the multi-label performances. Due to lack of space for data presentation, we have employed the numeric datasets only. Let *diff* be the difference in cardinalities of the majority and minority sets of a label. For each label, we have varied the cardinality of synthetic minority set between ($0.5 * diff$), ($1 * diff$), ($2 * diff$) and ($4 * diff$) and reported the results on Macro AUC and Macro F_1. Figs. 2 and 3 presents the results of Macro F_1 and Macro AUC respectively. Results indicate that Macro F_1 scores w.r.t minority (positive) classes of a dataset can improve or deteriorate on increasing the synthetic minority set cardinalities. The



(a) CAL500 (b) Emotions (c) RCV1 Subset1 (d) RCV1 Subset2

(e) Yeast

Fig. 3. Macro AUC results of the numeric datasets for various cardinalities of the synthetic minority set. The value in the x-axis is indicative of the cardinality of the $\mathbf{S}_{\min}$. Let *diff* be the difference between minority and majority sets. At x-axis value k, the cardinality of $\mathbf{S}_{\min}$ is $k \times diff$. k or x-axis co-ordinates are varied between 0.5, 1, 2 and 4.

results did not establish a fixed trend. On the contrary, Macro AUC scores have shown a negative trend with increasing number of synthetic minority points. Increasing the number of synthetic minority points beyond (1∗*diff*) causes over-learning of the minority class and interferes with the overall learning of the two classes and Macro AUC scores decreases.

7. Conclusion

In this paper, we have presented a novel oversampling scheme for multi-label, class-imbalanced datasets. The oversampling scheme is based on the distribution and density adaptive reverse k-nearest neighborhood exploration. The results confirm the efficaciousness of the proposed scheme to handle differential imbalance ratios of multi-label datasets against existing oversamplers and multi-label learners. The proposed scheme has been presented in a single-order framework. Future works of ours will focus on extending it to second and higher orders to facilitate label correlations in the oversampling phase. Currently, we are working towards designing a reverse nearest neighborhood based undersampler for majority classes of a multi-label dataset.

Declaration of competing interest

Indian Statistical Institute, Kolkata.

References

[1] S. Godbole, S. Sarawagi, Discriminative methods for multi-labeled classification, in: In Proceedings of the 8th Pacific-Asia Conference on Knowledge Discovery and Data Mining, 2004, pp. 22–30.
[2] T. Li, M. Ogihara, Toward intelligent music information retrieval, Multim. IEEE Trans. 8 (3) (2006) 564–574.
[3] N. Uvarov, Multi-label Classification of a Real-world Image Dataset, 2017.
[4] N. Inoue, E. Simo-Serra, T. Yamasaki, H. Ishikawa, Multi-label fashion image classification with minimal human supervision, in: 2017 IEEE International Conference on Computer Vision Workshops (ICCVW), 2017, doi:10.1109/ICCVW.2017.265.
[5] Z. Barutcuoglu, R.E. Schapire, O.G. Troyanskaya, Hierarchical multi-label prediction of gene function, Bioinformatics 22 (7) (2006) 830–836.
[6] S. Wang, X. Chang, X. Li, G. Long, L. Yao, Q.Z. Sheng, Diagnosis code assignment using sparsity-based disease correlation embedding, IEEE Trans. Knowl. Data Eng. 28 (12) (2016) 3191–3202.
[7] M.-L. Zhang, Y.-K. Li, X.-Y. Liu, Towards class-imbalance aware multi-label learning., in: IJCAI, 2015, pp. 4041–4047.
[8] Z.A. Daniels, D.N. Metaxas, Addressing imbalance in multi-label classification using structured hellinger forests., in: AAAI, 2017, pp. 1826–1832.
[9] A. Fernández, S. del Río, N.V. Chawla, F. Herrera, An insight into imbalanced big data classification: outcomes and challenges, Complex Intell. Syst. 3 (2) (2017) 105–120.
[10] G. Madjarov, D. Kocev, D. Gjorgjevikj, S. Džeroski, An extensive experimental comparison of methods for multi-label learning, Pattern Recognit. 45 (9) (2012) 3084–3104.
[11] M.S. Sorower, A Literature Survey on Algorithms for Multi-label Learning, Oregon State University, Corvallis, 2010.
[12] A comparison of multi-label feature selection methods using the problem transformation approach, Electron. Notes Theor. Comput. Sci. 292 (2013) 135–151.
[13] J. Fürnkranz, E. Hüllermeier, E. Loza Mencía, K. Brinker, Multilabel classification via calibrated label ranking, Mach. Learn. 73 (2) (2008) 133–153.
[14] S. hyeun Park, J. Fürnkranz, Efficient pairwise classification, in: ECML 2007. LNCS (LNAI, Springer, 2007, pp. 658–665.
[15] G. Tsoumakas, I. Katakis, I. Vlahavas, Random k-labelsets for multilabel classification, IEEE Trans. Knowl. Data Eng. 23 (7) (2011) 1079–1089.
[16] J. Read, B. Pfahringer, G. Holmes, E. Frank, Classifier chains for multi-label classification, Mach. Learn. 85 (3) (2011) 333–359.
[17] J. Read, L. Martino, D. Luengo, Efficient monte carlo optimization for multi-label classifier chains (2013) 3457–3461.
[18] W. Cheng, E. Hüllermeier, Combining instance-based learning and logistic regression for multilabel classification, Mach. Learn. 76 (2–3) (2009) 211–225.
[19] M.-L. Zhang, Z.-H. Zhou, Ml-knn: a lazy learning approach to multi-label learning, Pattern Recognit. 40 (7) (2007) 2038–2048.
[20] M. Zhang, Z. Zhou, Multi-label neural networks with applications to functional genomics and text categorization, IEEE Trans. Knowl. Data Eng. 18 (2006) 1338–1351.
[21] M.-L. Zhang, J.M. Peña, V. Robles, Feature selection for multi-label Naive Bayes classification, Inf. Sci. 179 (19) (2009) 3218–3229.
[22] M.-L. Zhang, L. Wu, Lift: multi-label learning with label-specific features, Pattern Anal. Mach. Intell. IEEE Trans. 37 (1) (2015).
[23] Z. Lin, G. Ding, M. Hu, J. Wang, Multi-label classification via feature-aware implicit label space encoding,
[24] F. Tai, H.-T. Lin, Multilabel classification with principal label space transformation, Neural Comput. 24 (9) (2012) 2508–2542.
[25] J. Huang, G. Li, Q. Huang, X. Wu, Joint feature selection and classification for multilabel learning, IEEE Trans. Cybern. 48 (3) (2018) 876–889.
[26] Granular multi-label feature selection based on mutual information, Pattern Recognit. 67 (2017) 410–423.
[27] C. Luo, L. Yu, E. Yang, H. Zhou, P. Ren, A benchmark image dataset for industrial tools, Pattern Recognit. Lett. (2019), doi:10.1016/j.patrec.2019.05.011.
[28] Y.-P. Wu, H.-T. Lin, Progressive random k-labelsets for cost-sensitive multi-label classification, Mach. Learn. 106 (2017) 671–694.
[29] K.-H. Huang, H.-T. Lin, Cost-sensitive label embedding for multi-label classification, Mach. Learn. 106 (9–10) (2017) 1725–1746.

[30] B. Goldluecke, D. Cremers, Convex relaxation for multilabel problems with product label spaces, in: K. Daniilidis, P. Maragos, N. Paragios (Eds.), Computer Vision – ECCV 2010, Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.
[31] H. Su, J. Rousu, Multilabel classification through random graph ensembles, Mach. Learn. 99 (2) (2015).
[32] M. Radovanović, A. Nanopoulos, M. Ivanović, Reverse nearest neighbors in unsupervised distance-based outlier detection, IEEE Trans. Knowl. Data Eng. 27 (5) (2015) 1369–1382.
[33] D. Pokrajac, A. Lazarevic, L.J. Latecki, Incremental local outlier detection for data streams, in: 2007 IEEE Symposium on Computational Intelligence and Data Mining, 2007, pp. 504–515.
[34] Y. Tao, M.L. Yiu, N. Mamoulis, Reverse nearest neighbor search in metric spaces, IEEE Trans. Knowl. Data Eng. 18 (9) (2006) 1239–1252.
[35] K. Ning, H.K. Ng, S. Srihari, H.W. Leong, A.I. Nesvizhskii, Examination of the relationship between essential genes in ppi network and hub proteins in reverse nearest neighbor topology, BMC Bioinf. 11 (2010) 505.
[36] C. Ji, H. Hu, Y. Xu, Y. Li, W. Qu, Efficient multi-dimensional spatial RKNN query processing with mapreduce, in: 2013 8th ChinaGrid Annual Conference, 2013, pp. 63–68.
[37] J. Lu, Y. Lu, G. Cong, Reverse spatial and textual k nearest neighbor search, in: SIGMOD Conference, 2011.
[38] A. Bryant, K. Cios, Rnn-dbscan: a density-based clustering algorithm using reverse nearest neighbor density estimates, IEEE Trans. Knowl. Data Eng. 30 (6) (2018) 1109–1121.
[39] D. Arthur, S.Y. Oudot, Reverse nearest neighbors search in high dimensions using locality-sensitive hashing, CoRR (2010). arXiv:1011.4955.
[40] X. Liu, C. Deng, B. Lang, D. Tao, X. Li, Query-adaptive reciprocal hash tables for nearest neighbor search, IEEE Trans. Image Process. 25 (2) (2016) 907–919.
[41] X. Liu, Z. Li, C. Deng, D. Tao, Distributed adaptive binary quantization for fast nearest neighbor search, IEEE Trans. Image Process. 26 (11) (2017) 5324–5336.
[42] N.V. Chawla, K.W. Bowyer, L.O. Hall, W.P. Kegelmeyer, Smote: synthetic minority over-sampling technique, J. Artif. Intell. Res. 16 (2002) 321–357.
[43] G. Tsoumakas, E. Spyromitros-Xioufis, J. Vilcek, I. Vlahavas, Mulan: a java library for multi-label learning, J. Mach. Learn. Res. 12 (2011) 2411–2414.
[44] J. Read, P. Reutemann, B. Pfahringer, G. Holmes, MEKA: a multi-label/multi-target extension to Weka, J. Mach. Learn. Res. 17 (21) (2016) 1–5.
[45] H. He, Y. Bai, E.A. Garcia, S. Li, Adasyn: adaptive synthetic sampling approach for imbalanced learning, in: 2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence), 2008.