Knowledge-Based Systems 192 (2020) 105292

Contents lists available at ScienceDirect

Knowledge-Based Systems

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

Dealing with class imbalance in classifier chains via random undersampling☆

Bin Liu *, Grigorios Tsoumakas

School of Informatics, Aristotle University of Thessaloniki, Thessaloniki, Greece

ARTICLE INFO

Article history:
Received 4 June 2019
Received in revised form 24 September 2019
Accepted 27 November 2019
Available online 4 December 2019

Keywords:
Multi-label learning
Class imbalance
Classifier chains
Undersampling

ABSTRACT

Class imbalance is an intrinsic characteristic of multi-label data. Most of the labels in multi-label data sets are associated with a small number of training examples, much smaller compared to the size of the data set. Class imbalance poses a key challenge that plagues most multi-label learning methods. Ensemble of Classifier Chains (ECC), one of the most prominent multi-label learning methods, is no exception to this rule, as each of the binary models it builds is trained from all positive and negative examples of a label. To make ECC resilient to class imbalance, we first couple it with random undersampling. We then present two extensions of this basic approach, where we build a varying number of binary models per label and construct chains of different sizes, in order to improve the exploitation of majority examples with approximately the same computational budget. Experimental results on 16 multi-label datasets demonstrate the effectiveness of the proposed approaches in a variety of evaluation metrics.

1. Introduction

Class imbalance is an intrinsic characteristic of multi-label data. Each training example in a multi-label dataset is typically associated with a small number of labels, much smaller than the total number of labels. This results in a sparse output matrix, where a small total number of positive class values is shared by a much larger number of example-label pairs. Though the distribution of the number of positive class values is not uniform across labels — in some real-world applications it follows a power law [1] — most of the labels are typically associated with a small number of positive class values. The imbalance ratio (*ImR*) of a label is the ratio of the number of the majority class examples to the number of minority class examples. Fig. 1(a) shows a density estimation plot and Fig. 1(b) a box-plot of the imbalance ratio logarithm for all labels in the 16 multi-label datasets of Table 1 that are part of our empirical study. We can see indeed that most of the labels are characterized by severe class imbalance.

The starting point of this work is *Ensemble of Classifier Chains* (ECC) [2], a popular multi-label learning algorithm with state-of-the-art predictive performance that is also accompanied by theoretical interpretation based on probability theory [3]. ECC suffers from class imbalance, as each of the binary models it builds is trained from all positive and negative examples of a label. While several approaches have been recently proposed to highlight and address the class imbalance problem in the context of multi-label learning, none of them has considered to build on top of ECC.

To make ECC resilient to class imbalance we contribute a new approach, called ECCRU, that couples it with random undersampling [4]. We then present two extensions of this basic approach, called ECCRU2 and ECCRU3, in order to improve the exploitation of majority class examples with approximately the same computational budget. This is achieved by building a varying number of binary models per label and constructing chains of different sizes. Experimental results on 16 multi-label datasets demonstrate the effectiveness of the proposed approaches in a variety of evaluation metrics.

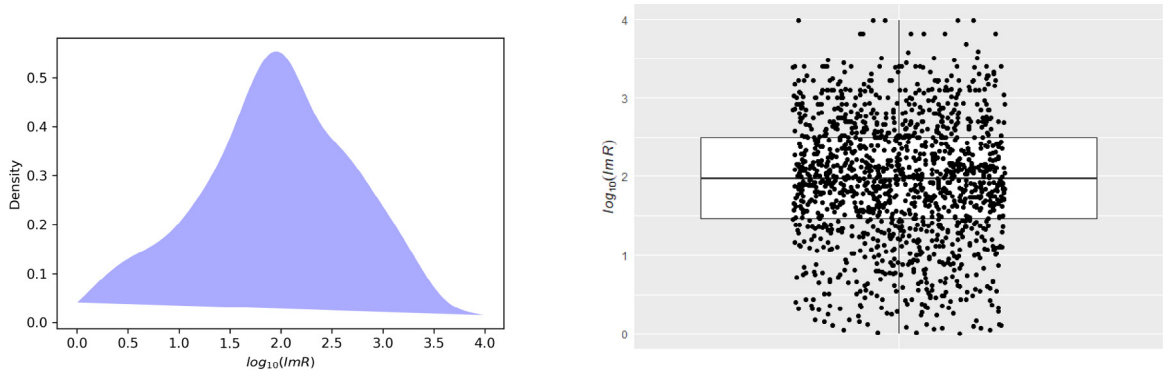
This paper extends our previous work [5] in the following main aspects:

- We present empirical results on the influence that the two parameters, θ_{min} and θ_{max}, of ECCRU3 have in its accuracy and training time.
- We discuss the special case, where the positive class, instead of the negative class, is associated with the majority of the examples.
- We compare our approach with an additional recent state-of-the-art method for dealing with class imbalance in multi-label learning [6].

☆ No author associated with this paper has disclosed any potential or pertinent conflicts which may be perceived to have impending conflict with this work. For full disclosure statements refer to https://doi.org/10.1016/j.knosys.2019.105292.

* Corresponding author.
E-mail addresses: binliu@csd.auth.gr (B. Liu), greg@csd.auth.gr (G. Tsoumakas).

https://doi.org/10.1016/j.knosys.2019.105292
0950-7051/



(a) (b)

Fig. 1. (a) Gaussian kernel density estimation plot and (b) box-plot (values superimposed and jittered) of the imbalance ratios of all labels in the 16 datasets of Table 1.

- We discuss related work in more detail.
- We exemplify the generation of partial chains in ECCRU2 to improve the clarity of its presentation.

The remainder of this article is organized as follows. Section 2 presents our basic approach, along with its two extensions. Section 3 discusses existing work related to dealing with class imbalance in the context of multi-label learning. Section 4 presents and discusses the experimental results. Finally, Section 5 summarizes the main contributions of this work and points to future directions.

2. Our approach

We first introduce the notation used in the rest of the paper and describe the ECC algorithm. Then, we present our approach for making classifier chains resilient to class imbalance along with two extensions that improve the exploitation of majority examples. In the last subsection, we analyze the computational complexity of the proposed methods.

2.1. Notation

Let $\mathcal{X} = \mathbb{R}^d$ be a d-dimensional input feature space, $L = \{l_1, l_2, \ldots, l_q\}$ a label set containing q labels and $\mathcal{Y} = \{0, 1\}^q$ a q-dimensional label space. $D = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)|1 \leqslant i \leqslant n\}$ is a multi-label training data set containing n instances. Each instance $(\boldsymbol{x}_i, \boldsymbol{y}_i)$ consists of a feature vector $\boldsymbol{x}_i \in \mathcal{X}$ and a label vector $\boldsymbol{y}_i \in \mathcal{Y}$, where y_{ij} is the jth element of $\boldsymbol{y}_i$ and $y_{ij} = 1(0)$ denotes that l_j is (not) associated with ith instance. For label l_j, $m_j = \min(|D_j^0|, |D_j^1|)$ and $M_j = \max(|D_j^0|, |D_j^1|)$ denote the number of minority and majority class examples respectively, where $D_j^b = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)|y_{ij} = b, 1 \leqslant i \leqslant n\}$. $ImR_j = M_j/m_j$ is the imbalance ratio of l_j. A multi-label method learns a mapping function $h : \mathcal{X} \rightarrow \{0, 1\}^q$ or $f : \mathcal{X} \rightarrow \mathbb{R}^q$ from D that given an unseen instance $\boldsymbol{x}$, outputs a label or a real-valued vector $\hat{\boldsymbol{y}}$ with the predicted labels of $\boldsymbol{x}$ or the corresponding relevance degrees to $\boldsymbol{x}$ respectively.

2.2. Ensemble of classifier chains

Classifier Chain (CC) is a well-known multi-label learning method that is based on the idea of chaining binary models [2]. CC exploits high-order label correlations by sequentially constructing one binary classifier for each label based on a chain (permutation) of the labels CH, where CH_j is the index of the label in L. The jth classifier h_j is constructed by the binary dataset whose class is label l_{CH_j} and the feature space of training instances is extended with the values of the previous labels in the chain. Once the classifier chain $\{h_1, \ldots, h_q\}$ is built, the unseen instance $\boldsymbol{x}$ is predicted by traversing all classifiers iteratively. The input of h_j is the $\boldsymbol{x}$ augmented by predictions of all preceding labels obtained from previous classifiers.

The performance of CC is highly affected by the sequence of the labels within the chain. To relieve the impact of label ordering and make the model more robust, the ECC algorithm constructs c different chains and corresponding CC models [2]. To make these models more diverse, each chain is trained on a different training set D' obtained by sampling with replacement ($|D'| = |D|$). The prediction of ECC for a test instance is obtained by combining the predictions of all CCs with the voting strategy. The jth element of relevance degree vector $\hat{\boldsymbol{y}}$, denoted by $\hat{y}_j$, is calculated as the number of CCs that predicts l_j as the relevant label of $\boldsymbol{x}$ divided by the number of chains c.

2.3. Ensemble of classifier chains with random undersampling

To deal with the class imbalance inherent in multi-label data, we firstly propose coupling CC with random undersampling [4], in order to balance the class distribution of each binary training set. This leads to the Classifier Chain with Random UnderSampling approach (CCRU), whose pseudo-code is shown in Algorithm 1.

Then CCRU builds binary classifiers sequentially according to a given label sequence. Random undersampling of majority examples is applied to each binary training set before building the corresponding classifier (line 4). In specific, m_j majority class examples are randomly sampled without replacement from label l_j in order to create a fully balanced binary training set.

In the original CC model, the true values of the labels are considered when using them as input features. Recent work found that two alternative approaches lead to better results in the context of multi-target regression chains [7]: (i) using in-sample estimates of the values of these labels by considering the predictions of the corresponding binary models on the training set, (ii) using out-of-sample estimates of the values of these labels by considering the cross-validated predictions of the corresponding binary models on the training set. CCRU avoids the second approach because cross-validation would construct training sets that are further deprived of the already small number of minority class examples, leading to a deviant distribution of predictions compared to the predictions of the corresponding binary models. In addition, cross-validation is very time consuming. Instead, CCRU follows the first of the above approaches, i.e. it considers the predictions of the corresponding binary models on the training set. As only a subset of the majority examples of the training

Algorithm 1: Training of CCRU

input : multi-label data set: D, sequence of labels: CH
output: CCRU model: $h = \{h_1, ..., h_{|CH|}\}$

1 $D_{CH_1} \leftarrow \{(\boldsymbol{x}_1, y_{1CH_1}), ..., (\boldsymbol{x}_{|D|}, y_{|D|CH_1})\}$;
2 $h \leftarrow \emptyset$;
3 **for** $j \leftarrow 1$ **to** $|CH|$ **do**
4 $\quad D^*_{CH_j} \leftarrow$ `RandomUnderSample`(D_{CH_j}) ; `/* apply random undersampling to` D_{CH_j} `*/`
5 $\quad$ train h_j based on $D^*_{CH_j}$;
6 $\quad h \leftarrow h \cup h_j$;
7 $\quad$ **if** $j < |CH|$ **then**
8 $\quad\quad D_{CH_{j+1}} \leftarrow \emptyset$;
9 $\quad\quad$ **foreach** $(\boldsymbol{x}, y)$ *in* D_{CH_j} **do**
10 $\quad\quad\quad \hat{y}_{CH_j} \leftarrow h_j(\boldsymbol{x})$;
11 $\quad\quad\quad \boldsymbol{x}' \leftarrow [x_1, ..., x_d, \hat{y}_{CH_1}, ..., \hat{y}_{CH_j}]$; `/* add augmented features */`
12 $\quad\quad\quad D_{CH_{j+1}} \leftarrow D_{CH_{j+1}} \cup (\boldsymbol{x}', y_{CH_{j+1}})$;
13 $\quad\quad$ **end**
14 $\quad$ **end**
15 **end**
16 **return** $h = \{h_1, ..., h_{|CH|}\}$;

Algorithm 2: Training of ECCRU

input : multi-label data set: D, number of chains: c
output: ECCRU model: $h = \{h^1, \ldots, h^c\}$

1 $L \leftarrow$ label set of D ;
2 **for** $i \leftarrow 1$ **to** c **do**
3 $\quad CH^i \leftarrow$ `RandomPermute`(L) ; `/* generate a chain by random permutation */`
4 $\quad D' \leftarrow$ `SampleWithReplacement`(D) ; `/* sample the` D `with replacement */`
5 $\quad h^i \leftarrow$ `TrainCCRU`(D', CH^i) ; `/* train a CCRU according to Algorithm 1 */`
6 **end**
7 $h \leftarrow \{h^1, \ldots, h^c\}$;
8 **return** $h = \{h^1, \ldots, h^c\}$;

Algorithm 3: Prediction of ECCRU

input : test instance $\boldsymbol{x}$, number of labels: q, ECCRU model: $h = \{h^1, h^2, ..., h^c\}$
output: relevance degree vector $\hat{\boldsymbol{y}}$

1 $\hat{\boldsymbol{y}} \leftarrow \boldsymbol{0}$;
2 **for** $i \leftarrow 1$ **to** c **do**
3 $\quad$ **for** $j \leftarrow 1$ **to** q **do**
4 $\quad\quad k \leftarrow$ the index of label trained by h^i_j ;
5 $\quad\quad \boldsymbol{x}' \leftarrow [x_1, ..., x_d, h^i_1(x), ..., h^i_{j-1}(x)]$;
6 $\quad\quad$ **if** $h^i_j(\boldsymbol{x}') = 1$ **then**
7 $\quad\quad\quad \hat{y}_k \leftarrow \hat{y}_k + 1$;
8 $\quad\quad$ **end**
9 $\quad$ **end**
10 **end**
11 **for** $j \leftarrow 1$ **to** q **do**
12 $\quad \hat{y}_j \leftarrow \hat{y}_j / c$;
13 **end**
14 **return** $\hat{\boldsymbol{y}}$;

set are used for the training of the corresponding binary model (line 5), CCRU essentially considers a mixture of in-sample and out-of-sample predictions: in-sample for the minority class and the equal number of retained majority class examples, and out-of-sample for the rest of the majority class examples that were removed (lines 7–14). Specifically, for each label, except for the last label $l_{|CH|}$, each instance $\boldsymbol{x}$ in D_{CH_j} is given as input to h_j to get its prediction $\hat{y}_{CH_j}$ (line 10). If $\boldsymbol{x}$ is in $D^*_{CH_j}$, then $\hat{y}_{CH_j}$ is an in-sample prediction, otherwise it is an out-of-sample prediction. Subsequently, the features of $\boldsymbol{x}$ and the corresponding predictions for previous labels are connected as new features $\boldsymbol{x}'$, which along with $y_{CH_{j+1}}$ constitute the instance $(\boldsymbol{x}', y_{CH_{j+1}})$ of $D_{CH_{j+1}}$ that would be used in the next iteration, where x_1 and x_d is the 1st and d^{th} feature of $\boldsymbol{x}$ respectively (lines 11–12).

Similar to ECC, the Ensemble of Classifier Chains with Random Undersampling (ECCRU) algorithm aggregates several CCRUs that are built upon different label sequences and resampled versions of the original training set. The training and prediction algorithms of ECCRU are presented in Algorithms 2 and 3, respectively.

2.4. Improving the exploitation of majority examples

In ECCRU, the probability that a majority example of a label is eventually used for training the binary models of that label depends on the number of minority, m, and majority, M, examples of that label, as well as on the number of chains, c. In each chain, sampling of all the training examples with replacement is first performed once, followed by separate samplings of the majority examples of each label without replacement. If we skip the first sampling process for the sake of simplifying the analysis, then the probability that a majority example of a label is selected in at least one of the c chains of ECCRU, denoted as P, can be obtained by Eq. (1).

$$P = 1 - \left(1 - \frac{m}{M}\right)^c \tag{1}$$

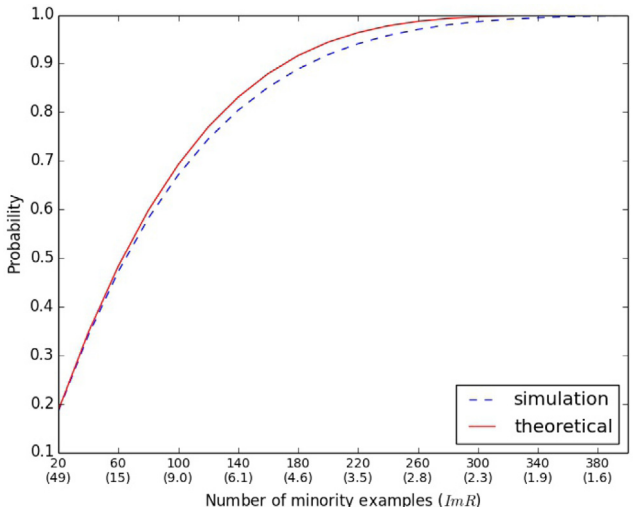


Fig. 2. The empirically estimated (simulated) and theoretically approximated probability of a majority example of a label being retained for training by at least one of the 10 corresponding models of ECCRU with 10 chains, assuming 1000 training examples and a number of minority examples varying from 20 to 400 with a step of 20. In the simulation, the sampling process was conducted 10,000 times.

Fig. 2 plots Eq. (1) for 10 chains, 1000 training examples and varying number of minority samples, as well as the empirical probability in question estimated using 10,000 runs. We notice that for $ImR > 15$ this probability is less than 0.5, with an alternative interpretation being that less than half of the majority examples are eventually used by ECCRU in such a case. As intuitively expected, we see that the higher the ImR of a label, the lower the exploitation of its majority examples.

A straightforward way to increase the exploitation of majority examples when ImR is high is to increase the number of chains.

$Chain_1$	l_1	l_2	l_3	l_4	$Chain_7$	l_1	l_2	l_3
$Chain_2$	l_3	l_1	l_2	l_4	$Chain_8$	l_3	l_2	l_1
$Chain_3$	l_1	l_2	l_4	l_3	$Chain_9$	l_1	l_2	
$Chain_4$	l_2	l_4	l_3	l_1	$Chain_{10}$	l_2	l_1	
$Chain_5$	l_2	l_1	l_3	l_4	$Chain_{11}$	l_2	l_1	
$Chain_6$	l_4	l_3	l_2	l_1	$Chain_{12}$	l_1	l_2	

Fig. 3. The example of ECCRU2 to constructing chains with various length on a dataset with 100 instances and 4 labels, each with 10, 20, 30 and 40 minority instances respectively.

This is also theoretically grounded based on Eq. (1). Increasing the number of chains however leads to increased computational cost. We instead consider a variation of our algorithm that improves the exploitation of majority examples without increasing the computational budget. A key observation is that each label contributes a different computational cost to ECCRU, which is proportional to the number of its minority examples, as each corresponding classifier is trained with twice that number of examples. For example, given a dataset with 100 training examples and 4 labels (l_1, l_2, l_3, l_4), each with 10, 20, 30 and 40 minority examples respectively, the binary classifiers of l_1, l_2, l_3 and l_4 will be trained with 20, 40, 60 and 80 training examples respectively.

Our proposal is to redistribute this computational cost by building a different number of classifiers per label, inversely proportional to its number of minority examples. This way we can achieve uniform exploitation of majority examples across labels at the same computational cost. We call this variation of our approach ECCRU2. Continuing the previous example, if we build 10 chains, then the total number of exploited majority examples is 1000 (10 times 10 + 20 + 30 + 40). Our approach divides this computational budget equally across labels, i.e. 250 majority examples per label. We then divide this with the number of minority examples of each label to get the number of classifiers to build for each label, i.e. 25, 12.5, $8.\overline{3}$ and 6.25. In general, given q labels and a budget of c chains, the number of classifiers, c_j, constructed by ECCRU2 for label j is given by Eq. (2).

$$c_j = \lfloor \frac{c \sum_{k=1}^{q} m_k}{q m_j} \rfloor \tag{2}$$

To accommodate the fact that the number of classifiers to be constructed differs among labels, ECCRU2 considers partial chains containing a decreasing number of labels until the minimum of two labels. Continuing the aforementioned example, ECCRU2 would build 6 chains with all four labels, 2 chains including the first three labels and 4 chains containing only l_1 and l_2. An example of such chains is illustrated in Fig. 3. Although the same short-length chains may appear more than once, the initial process of sampling with replacement and, most importantly, the random undersampling process that follows lead to different majority class instances being used to train the models, and eventually to diverse models.

The pseudo-code of the training process of ECCRU2 is given in Algorithm 4. Firstly, the number of classifiers trained for each label c_j is calculated according to Eq. (2). To limit the number of classifiers in the case of highly imbalanced labels, we confine c_j to be less than a predefined maximal value c_{max}, defined as a multiple of c: $c_{max} = c\theta_{max}$ (line 3). In our empirical study we set $c = 10$, $\theta_{max} = 10$ and therefore $c_{max} = 100$. Then, in each iteration of building a CCRU model, labels whose corresponding counter cn_j recording the number of classifier needed to be trained is larger than 0 are added into the label set S, and only labels collected in S are utilized to generate the label sequence to train the current CCRU model. The loop (in line 6–21) terminates when c_{max} chains have been built or $|S| < 2$. The rest parts of the training phase of ECCRU2 are identical to ECCRU.

The pseudo-code of the prediction process of ECCRU2 is given in Algorithm 5. In ECCRU2, the number of binary classifiers contained in CCRU h^i, denoted as $|h^i|$, does not always equal q. Hence, a q dimensional vector $\boldsymbol{cc}$ is introduced to count the number of binary classifiers for each label, which is used in line 14 to normalize the $\hat{y}_j$, for $j = 1, \ldots, q$. The rest parts of prediction process of ECCRU2 are as in ECCRU.

One issue in ECCRU2 is that very few classifiers, even just one, can be built in the case of balanced labels with large m_j, leading to fewer full-sized chains being built. To address this problem, a variant of ECCRU2 called ECCRU3 is proposed. The only change in ECCRU3 is the addition of a lower bound c_{min} for c_j, where $c_{min} = c\theta_{min}$ and $\frac{1}{c} \leqslant \theta_{min} \leqslant 1$ to ensure that $1 \leqslant c_{min} \leqslant c$. Hence, the confined c_j (cn_j) is computed as $\min\{\max\{c_j, c\theta_{min}\}, c\theta_{max}\}$ and so at least c_{min} chains containing all of the labels are built. In our empirical study we set $c = 10$, $\theta_{min} = 0.5$ and therefore $c_{min} = 5$. The rest parts of the training and prediction process of ECCRU3 are the same with ECCRU2.

Note that when $\theta_{min} = 1/c$ (e.g. 0.1 when $c = 10$), ECCRU3 reverts to ECCRU2, as the minimal number of binary classifier built for each label is 1. Furthermore, when $\theta_{min} = 1$ and $\theta_{max} = 1$, ECCRU3 reverts to ECCRU, as to the lower and upper boundaries of the number of binary classifiers for each label are always c.

Algorithm 4: Training of ECCRU2

input : multi-label data set: D, number of labels: q, standard number of chains: c, the coefficient of maximal number of chains: θ_{max}
output: ECCRU2 model: $h = \{h^1, ..., h^{c'}\}$

1 **for** $j \leftarrow 1$ **to** q **do**
2 calculate c_j according to Eq. (2) ;
3 $c_j \leftarrow \min\{c_j, c\theta_{max}\}$;
4 $cn_j \leftarrow c_j$; /* the number of classifiers needed to be built for each label */
5 **end**
6 $c' \leftarrow 0$; /* the counter to record the number of chains built actually */
7 **for** $i \leftarrow 1$ **to** $c\theta_{max}$ **do**
8 $S \leftarrow \emptyset$;
9 **for** $j \leftarrow 1$ **to** q **do**
10 **if** $cn_j > 0$ **then**
11 $S \leftarrow S \cup l_j$;
12 $cn_j \leftarrow cn_j - 1$;
13 **end**
14 **end**
15 **if** $|S| < 2$ **then**
16 **break**;
17 **end**
18 $CH^i \leftarrow$ RandomPermute(S) ; /* generate a chain by random permutation */
19 $D' \leftarrow$ SampleWithReplacement(D) ; /* sample the D with replacement */
20 $h^i \leftarrow$ TrainCCRU(D', CH^i) ; /* train a CCRU according to Algorithm 1 */
21 $c' \leftarrow c' + 1$;
22 **end**
23 $h \leftarrow \{h^1, ..., h^{c'}\}$;
24 **return** $h = \{h^1, ..., h^{c'}\}$;

Algorithm 5: Prediction of ECCRU2

input : test instance $\boldsymbol{x}$, number of labels: q, ECCRU2 model: $h = \{h^1, h^2, \ldots, h^{c'}\}$
output: degree relevance vector $\hat{\boldsymbol{y}}$

$\hat{\boldsymbol{y}} \leftarrow \mathbf{0}$;
$\boldsymbol{cc} \leftarrow \mathbf{0}$; /* $\boldsymbol{cc}$ is a q dimensional counter */
for $i \leftarrow 1$ **to** c' **do**
  for $j \leftarrow 1$ **to** $|h^i|$ **do**
    $k \leftarrow$ the index of label trained by h^i_j ;
    $cc_k \leftarrow cc_k + 1$;
    $\boldsymbol{x}' \leftarrow [x_1, \ldots, x_d, h^i_1(x), \ldots, h^i_{j-1}(x)]$;
    if $h^i_j(\boldsymbol{x}') = 1$ **then**
      $\hat{y}_k \leftarrow \hat{y}_k + 1$;
    end
  end
end
for $j \leftarrow 1$ **to** q **do**
  $\hat{y}_j \leftarrow \hat{y}_j / cc_j$;
end
return $\hat{\boldsymbol{y}}$;

2.5. Complexity analysis

Let us define $\Theta_t(m_j, d)$ and $\Theta_p(d)$ the complexity of training and prediction of a binary classifier for label l_j, respectively. The complexity of ECCRU is $O\left(c\sum_{j=1}^{q}\Theta_t(m_j, d) + ncq\Theta_p(d)\right)$ for training and $O\left(cq\Theta_p(d)\right)$ for prediction. The training complexity of ECCRU2 is given in Eq. (3), while its prediction complexity is $O\left(\Theta_p(d)\sum_{j=1}^{q} c_j\right)$. In both algorithms, the first part of the training complexity concerns building classifiers and the second relates to generating the augmented feature space.

$$O\left(\frac{c}{q}\sum_{k=1}^{q} m_k * \sum_{j=1}^{q}\left(\frac{1}{m_j}\Theta_t(m_j, d)\right) + n\Theta_p(d)\sum_{j=1}^{q} c_j\right) \tag{3}$$

The number of binary classifiers in ECCRU2 are more than in ECCRU, which results in larger complexity of prediction and generation of augmented features. However, the comparison between the training complexity of the first part of ECCRU2 and ECCRU depends on the m_j and $\Theta_t(m_j, d)$ of each label. The formulation of the training and testing complexity of ECCRU3 is the same with ECCRU2, but ECCRU3 is more time-consuming than ECCRU2 in both processes in practice, because a larger lower bound in the number of classifiers is applied to ECCRU3.

3. Related work

The traditional class imbalance problem in binary and multi-class classification has been widely studied [8]. There are four main kinds of approaches to deal with it: (1) data level methods make the class distribution more balanced via resampling instances [9], (2) algorithm level methods modify the existing learning algorithm to handle class imbalance [10], (3) cost-sensitive learning methods bias the classifier to the minority class via assigning a higher misclassification cost to the minority class instance [11], and (4) ensemble methods are usually combined with data level and cost-sensitive learning methods, which incorporate various classifiers based on different misclassification costs or built on resampled datasets with diverse sampling ratios [12].

In multi-label learning, approaches to tackle class imbalance problem are more complex due to the multi-dimensional output space. A series of approaches by the same research group have been proposed for dealing with class imbalance in the context of multi-label learning using under/over-sampling. LP-RUS and LP-ROS are two twin sampling methods, of which the former removes instances assigned with most frequent labelset and the latter replicates instances whose labelset appears fewer times [13]. ML-RUS and ML-ROS delete instances with majority labels and clone examples with minority labels, respectively [14]. MLeNN is a heuristic undersampling method based on the Edited Nearest Neighbor (ENN) rule, which eliminates instances only with majority labels and similar labelset of its neighbors [15]. MLSMOTE tries to make a multi-label dataset more balanced via generating synthetic instances according to a randomly selected instance containing minority labels and its neighbors [16]. REMEDIAL decomposes each complex instance into two easier instances, one of which merely contains majority labels and another only with minority labels [17].

Another kind of methods deal with the imbalance problem of multi-label learning via transforming the multi-label dataset to several binary/multi-class classification problems. A simple strategy is dividing the multi-label dataset into several independent binary datasets, as BR does [18], and using sampling or ensemble strategy to solve each imbalanced binary classification problem [19–22]. These BR based methods totally ignore any label correlations. Cross-Coupling Aggregation (COCOA) [23] is proposed to leverage the exploitation of label correlations as well as the exploration of imbalance via building one binary-class imbalance learner and several multi-class imbalance learners for each label with the assistance of sampling. In the COCOA, each label couples with K labels and each label pair is utilized to train one multi-class classifier. The Sparse Oblique Structured Hellinger Forests (SOSHF) [6] transforms the multi-label learning task to an imbalanced single label classification assignment via cost-sensitive clustering method and the transformed imbalanced classification problem is solved by tree classifiers where splitting point is determined by minimizing the sparse Hellinger loss.

In addition, some approaches that extend existing multi-label learning methods to tackle class-imbalance problem have been proposed. In the data enrichment process of [24], a small balanced subset of the training data is selected to initialize a neural network model. This subset is incrementally updated by removing instances with low energy error and adding examples from the neighborhoods of existing examples with high energy error. IMIMLRB [25] is an improved multi-instance multi-label radial basis function neural network that deals with class imbalance by applying a data density based k-medoids algorithm to produce more balanced medoids for the first layer of the model and by using singular value decomposition to update the weights of the model. The 3D convolutional multi-label-k-output neural network model for action recognition in hockey videos [26] deals with imbalanced labels by adjusting the model weights and prediction threshold. CSRankSVM [27] is a cost sensitive rank support vector machine that addresses the imbalanced distribution of labelsets. CSRankSVM assigns larger weights to instances associated with low frequency labelsets and minimizes the weighted ranking loss. TSMLHN [28] is a two-stage multi-label hypernetwork that emphasizes imbalanced labels via introducing an imbalance-specific post-processing operation: the prediction probabilities of positive instance for highly imbalanced labels are increased based on the information provided by the co-occurrences among balanced and imbalanced labels.

Finally, other approaches employing different strategies for addressing the imbalance problem in multi-label learning, include RMLS [29], MMIB [30] and BPL [31]. RMLS [29] views the class imbalance problem from the point of the instances, noting that there are typically more irrelevant labels than relevant labels in each instance. It leverages an unbiased loss function which samples irrelevant labels to balance the contributions of relevant

Table 1
The 16 multi-label data sets used in this study. Columns *n*, *d*, *q* denote the number of instances, features and labels respectively, *LC* the label cardinality, *MeanImR* and *MaxImR* the average and maximum *ImR* of the labels and *CVImR* the normalized standard deviation of the *ImR* of the labels.

Dataset	Domain	*n*	*d*	*q*	*LC*	*MaxImR*	*MeanImR*	*CVImR*
bibtex	Text	7 395	183	159	2.402	144	87.7	0.4097
cal500	Music	502	68	174	26	99.4	22.3	1.129
corel5k	Image	5 000	499	347	3.517	2499	522	1.13
enron	Text	1 702	100	52	3.378	850	107	1.496
eurlex-sm	Text	19 348	500	186	2.213	9673	1056	1.849
flags	Image	194	19	7	3.392	6.462	2.753	0.7108
genbase	Biology	662	1186	24	1.248	330	78.8	1.286
mediamill	Video	43 907	120	101	4.376	1415	331	1.178
medical	Text	978	144	35	1.245	488	143	1.115
rcv1subset1	Text	6 000	472	101	2.88	2999	236	2.089
rcv1subset2	Text	6 000	472	101	2.634	1999	191	1.724
scene	Image	2 407	294	6	1.074	5.613	4.662	0.1485
tmc2007–500	Text	28 596	500	22	2.158	63.8	25.8	0.7909
yahoo-Arts1	Text	7 484	231	25	1.654	1246	101	2.468
yahoo-Business1	Text	11 214	219	28	1.599	3737	286	2.453
yeast	Biology	2 417	103	14	4.237	70.1	8.954	1.997

Table 2
Average rank of all methods in terms of five evaluation metrics and training time.

	BR	ECC	BRUS	EBRUS	COCOA	SOSHF	ECCRU	ECCRU2	ECCRU3
F-measure	7.63	3.91	7.75	5.13	4.13	5.44	4.44	3.81	**2.78**
G-mean	8.56	6.00	3.25	5.81	5.31	7.00	**2.69**	3.59	2.78
Balanced Acc.	8.69	6.00	6.13	5.44	4.44	6.88	2.25	2.97	**2.22**
AUC-ROC	8.81	6.31	7.94	4.09	4.69	4.88	3.19	3.09	**2.00**
AUC-PR	7.94	3.00	8.88	5.75	4.94	4.06	4.09	3.91	**2.44**
Training time	3.06	8.31	**1.00**	5.06	3.06	7.75	5.13	5.44	6.19

and irrelevant labels in each instance. It also samples different labels in each batch of a mini-batch stochastic gradient descent optimization process to make the model more robust. MMIB [30] addresses the problem of class imbalance together with the problem of missing labels by formulating multi-label learning as a constrained submodular minimization task. It handles class imbalance by introducing two class cardinality bounds, one that constrains the number of positive labels for each instance and another one that constrains the number of relevant instances for each label. BPL [31] starts from a matrix containing the distinct label vectors of the training set and increases its dimensionality by adding artificially constructed labels. These *pseudo-labels* are created so that the expanded matrix maximizes an objective function involving three components: distance between the rows of the matrix (differentiate label vectors), distance between the columns of the matrix (differentiate labels) and balance between positive and negative values (deal with class imbalance). For an unseen instance, the prediction is made by BPL based on binary classifiers built upon both original labels and pseudo-labels.

Compared to the above approaches, the strengths of the methods that we proposed are as follows. Firstly, they build on top of a theoretically grounded and highly accurate method, ECC. Secondly, they inherit the ability of ECC to model correlation among many labels, in contrast for example to [23] that is second-order and [19–22] that are first-order methods. Thirdly, they are algorithm independent, as they can be combined with any binary classifier that best fits the problem at hand, in contrast to [6,24–28] that build on top of particular learning paradigms.

4. Empirical analysis

We first introduce the setup of our experiments. Then we present a first set of experimental results along with significance tests. Subsequently we discuss how the different methods behave under different levels of imbalance ratio. We then analyze the effect that different parameter settings have on ECCRU3. Finally, we investigate the special case where the positive, instead of the negative, class is associated with the majority of the training examples.

4.1. Experimental setup

Our empirical study is based on 16 multi-label data sets obtained from Mulan's [32] GitHub repository.[1] Table 1 lists these datasets along with their main statistics. In textual data sets with more than 1000 features we applied a simple dimensionality reduction approach that retains the top 10% (bibtex, enron, eurlex-sm, medical) or top 1% (rcv1subset1, rcv1subset2, yahoo-Arts1, yahoo-Business1) of the features ordered by number of non-zero values (i.e. frequency of appearance), similar to [23]. Besides, we remove labels only containing one minority class instance, because when splitting the dataset into training and test set, there may be only majority class instances of those extremely imbalanced labels in training set.

The proposed approaches are compared against six multi-label learning methods. Two of them are imbalance agnostic ones, namely the Binary Relevance (BR) baseline [18] and the state-of-the-art ECC [2], on which the proposed approaches build. Three are imbalance aware ones that similarly to ours are based on random undersampling, namely BR with random undersampling (BRUS), ensemble of BRUS (EBRUS) and the state-of-the-art COCOA [23]. The last one, SOSHF [6], is another state-of-the-art method that is based on decision tree learning.

In ECCRU2 and ECCRU3, θ_{max} is set to 10. In ECCRU3, θ_{min} is set to 0.5. The ensemble size is set to 10 for all ensemble methods (ECC, EBRUS, ECCRU*, SOSHF). In COCOA the number of coupling class labels is set to $min(q-1, 10)$ as in [23]. The rest of the parameters of SOSHF are set as in [6]. A decision tree is used as the base classifier in all methods apart from SOSHF, which is based on its own decision tree implementation, minimizing the sparse Hellinger loss.

We employ five widely used binary metrics for imbalanced data [8,33]: F-measure, G-mean, Balanced Accuracy, area under the receiver operating characteristic curve (AUC-ROC) and area under the precision–recall curve (AUC-PR). The first three are

[1] https://github.com/tsoumakas/mulan/tree/master/data/multi-label.

Table 3
Results of the Wilcoxon signed rank test with Bergman-Hommel's correction at the 5% level among all pairs of methods. "↑" ("↓") denotes the method in bold typeface in the upper-left corner of each subtable is significantly superior (inferior) to the corresponding method of each row. "–" denotes lack of significant difference between the two methods. Abbreviations stand for: (F)-measure, (G)-mean, (B)alanced Accuracy, AUC-(R)OC, AUC-(P)R, Training (T)ime.

BR vs.	*F*	*G*	*B*	*A*	*P*	*T*	**ECC vs.**	*F*	*G*	*B*	*A*	*P*	*T*
ECC	↓	↓	↓	↓	↓	↑	BR	↑	↑	↑	↑	↑	↓
BRUS	–	↓	↓	↓	↑	↓	BRUS	↑	↓	–	↑	↑	↓
EBRUS	–	↓	↓	↓	↓	–	EBRUS	–	–	–	↓	↑	↓
COCOA	↓	↓	↓	↓	↓	–	COCOA	–	–	↓	↓	↑	↓
SOSHF	–	↓	↓	↓	↓	↑	SOSHF	–	↑	–	↓	–	–
ECCRU	↓	↓	↓	↓	↓	↑	ECCRU	–	↓	↓	↓	–	↓
ECCRU2	↓	↓	↓	↓	↓	↑	ECCRU2	–	↓	↓	↓	–	↓
ECCRU3	↓	↓	↓	↓	↓	↑	ECCRU3	–	↓	↓	↓	–	↓
BRUS vs.	*F*	*G*	*B*	*A*	*P*	*T*	**EBRUS vs.**	*F*	*G*	*B*	*A*	*P*	*T*
BR	–	↑	↑	↑	↓	↑	BR	–	↑	↑	↑	↑	–
ECC	↓	↑	–	↓	↓	↑	ECC	–	–	–	↑	↓	↑
EBRUS	↓	↑	–	↓	↓	↑	BRUS	↑	↓	–	↑	↑	↓
COCOA	↓	↑	↓	↓	↓	↑	COCOA	–	–	–	–	–	↓
SOSHF	–	↑	↑	↓	↓	↑	SOSHF	–	↑	↑	–	–	–
ECCRU	↓	–	↓	↓	↓	↑	ECCRU	–	↓	↓	–	↓	–
ECCRU2	↓	–	↓	↓	↓	↑	ECCRU2	↓	↓	↓	–	↓	–
ECCRU3	↓	–	↓	↓	↓	↑	ECCRU3	↓	↓	↓	↓	↓	–
COCOA vs.	*F*	*G*	*B*	*A*	*P*	*T*	**SOSHF vs.**	*F*	*G*	*B*	*A*	*P*	*T*
BR	↑	↑	↑	↑	↑	–	BR	–	↑	↑	↑	↑	↓
ECC	–	–	↑	↑	↓	↑	ECC	–	↓	–	↑	–	–
BRUS	↑	↓	↑	↑	↑	↓	BRUS	–	↓	↓	↑	↑	↓
EBRUS	–	–	–	–	–	↑	EBRUS	–	↓	↓	–	–	–
SOSHF	–	↑	↑	–	–	↑	COCOA	–	↓	↓	–	–	↓
ECCRU	–	↓	↓	↓	–	↑	ECCRU	–	↓	↓	↓	–	–
ECCRU2	–	↓	↓	↓	–	↑	ECCRU2	–	↓	↓	↓	–	↓
ECCRU3	–	↓	↓	↓	↓	↑	ECCRU3	–	↓	↓	↓	–	–
ECCRU vs.	*F*	*G*	*B*	*A*	*P*	*T*	**ECCRU2 vs.**	*F*	*G*	*B*	*A*	*P*	*T*
BR	↑	↑	↑	↑	↑	↓	BR	↑	↑	↑	↑	↑	↓
ECC	–	↑	↑	↑	–	↑	ECC	–	↑	↑	↑	–	↑
BRUS	↑	–	↑	↑	↑	↓	BRUS	↑	–	↑	↑	↑	↓
EBRUS	–	↑	↑	–	↑	–	EBRUS	↑	↑	↑	–	↑	–
COCOA	–	↑	↑	↑	–	↓	COCOA	–	↑	↑	↑	–	↓
SOSHF	–	↑	↑	↑	–	–	SOSHF	–	↑	↑	↑	–	↑
ECCRU2	–	–	–	–	–	–	ECCRU	–	–	–	–	–	–
ECCRU3	↓	–	–	↓	↓	–	ECCRU3	↓	↓	↓	↓	↓	↑
ECCRU3 vs.	F	G	B	A	P	T							
BR	↑	↑	↑	↑	↑	↓							
ECC	–	↑	↑	↑	–	↑							
BRUS	↑	–	↑	↑	↑	↓							
EBRUS	↑	↑	↑	↑	↑	–							
COCOA	–	↑	↑	↑	↑	↓							
SOSHF	–	↑	↑	↑	–	–							
ECCRU	↑	–	–	↑	↑	–							
ECCRU2	↑	↑	↑	↑	↑	↓							

Table 4
The summary of statistical test. The n_1/n_2 denotes the corresponding method is significantly superior to n_1 methods and inferior to n_2 methods. Abbreviations are same with Table 3.

	F	*G*	*B*	*A*	*P*	*T*
BR	0/5	0/8	0/8	0/8	1/7	5/1
ECC	2/0	2/4	1/4	2/6	4/0	0/7
BRUS	0/6	5/0	2/4	1/7	0/8	**8/0**
EBRUS	1/2	2/4	2/3	3/1	2/4	1/2
COCOA	2/0	2/4	4/3	3/3	2/2	6/1
SOSHF	0/0	1/7	1/6	3/3	2/0	0/4
ECCRU	2/1	5/0	6/0	5/1	3/1	1/3
ECCRU2	3/1	5/1	6/1	5/1	3/1	3/3
ECCRU3	**5/0**	**6/0**	**7/0**	**8/0**	**6/0**	1/4

computed on top of binary predictions, while the last two on top of ranked lists of test instances in order of relevance to the positive class, which most of the times is the minority class. For all these metrics, the higher their value, the better the accuracy of the corresponding algorithm. Binary predictions are obtained after setting a separate threshold $t \in \{0, 0.05, \ldots, 1\}$ per label. This threshold is set so as to maximize the corresponding evaluation metric (F-measure, G-mean, Balanced Accuracy) on the training set.

We compute the average of the above metrics across all labels, an approach to aggregating binary measures in multi-class and multi-label tasks that is called *macro-averaging*. The alternative approach, *micro-averaging*, collects the predictions for all labels as if they were part of a single binary classification task. Macro-averaging is more suitable for imbalanced learning as it treats all labels equally, in contrast to micro-averaging where the contribution of each label depends on the frequency of the positive class [34]. We apply 5 × 2-fold cross validation with multi-label stratification [35] to each dataset and the average results are reported.

We used the implementations of BR and ECC from Mulan [32] and the Matlab implementation of SOSHF provided by the authors.[2] BRUS, EBRUS, COCOA and our ECCRU approaches were also implemented in Mulan. The experiments were conducted on a machine with 4 × 10-core CPUs running at 2.27 GHz.

4.2. Results and significance tests

Table 2 shows the average rank of each method in terms of the five evaluation metrics plus the training time, with the best method highlighted with bold typeface. Detailed results are listed in Tables A.1–A.6 of Appendix. We first notice that ECCRU3 achieves the best results in all evaluation metrics, with the exception of G-mean, where it is second best behind ECCRU. In addition, the proposed methods achieve the top 3 positions for all evaluation metrics, with the exception of F-measure and AUC-PR where ECC achieves the third and second position respectively, and G-mean where BRUS is the third best method. In terms of training time, BRUS achieves the best results followed by BR and COCOA. The proposed methods come next, followed by SOSHF. ECC is the slowest method.

To examine the statistical significance of the differences among the methods participating in our empirical study, we employ the Friedman test, followed by the Wilcoxon signed rank test with Bergman-Hommel's correction at the 5% level, following literature guidelines [36,37]. Tables 3 and 4 present the results. We notice that ECCRU3 achieves the most significant wins than any other method in all measures: 5 in F-measure, 6 in G-mean and AUC-PR, 7 in Balanced Accuracy, 8 in AUC-ROC. ECC, COCOA and SOSHF in F-measure, BRUS in G-mean, and ECC along with SOSHF in AUC-PR are the only 6 out of 30 cases of non-significant difference between ECCRU3 and the six competing methods in the five evaluation measures. In terms of training time, BRUS has the most wins minus losses (7), followed by COCOA (5), BR (4), ECCRU2 (0), EBRUS (-1), ECCRU (-2), ECCRU3 (-3), SOSHF (-4) and ECC (-7). While ECC is competitive with ECCRU3 in F-measure and AUC-PR, it is significantly worse in training time. If G-mean (F-measure) is the measure of interest, then BRUS (COCOA) is an algorithm to consider as it is both highly accurate and efficient. Besides, there is no significant difference between the training times of ECCRU and ECCRU3, which verifies the accomplishment of our purpose that improving the performance without additional computational cost.

[2] https://github.com/ZDanielsResearch/SOSHF.

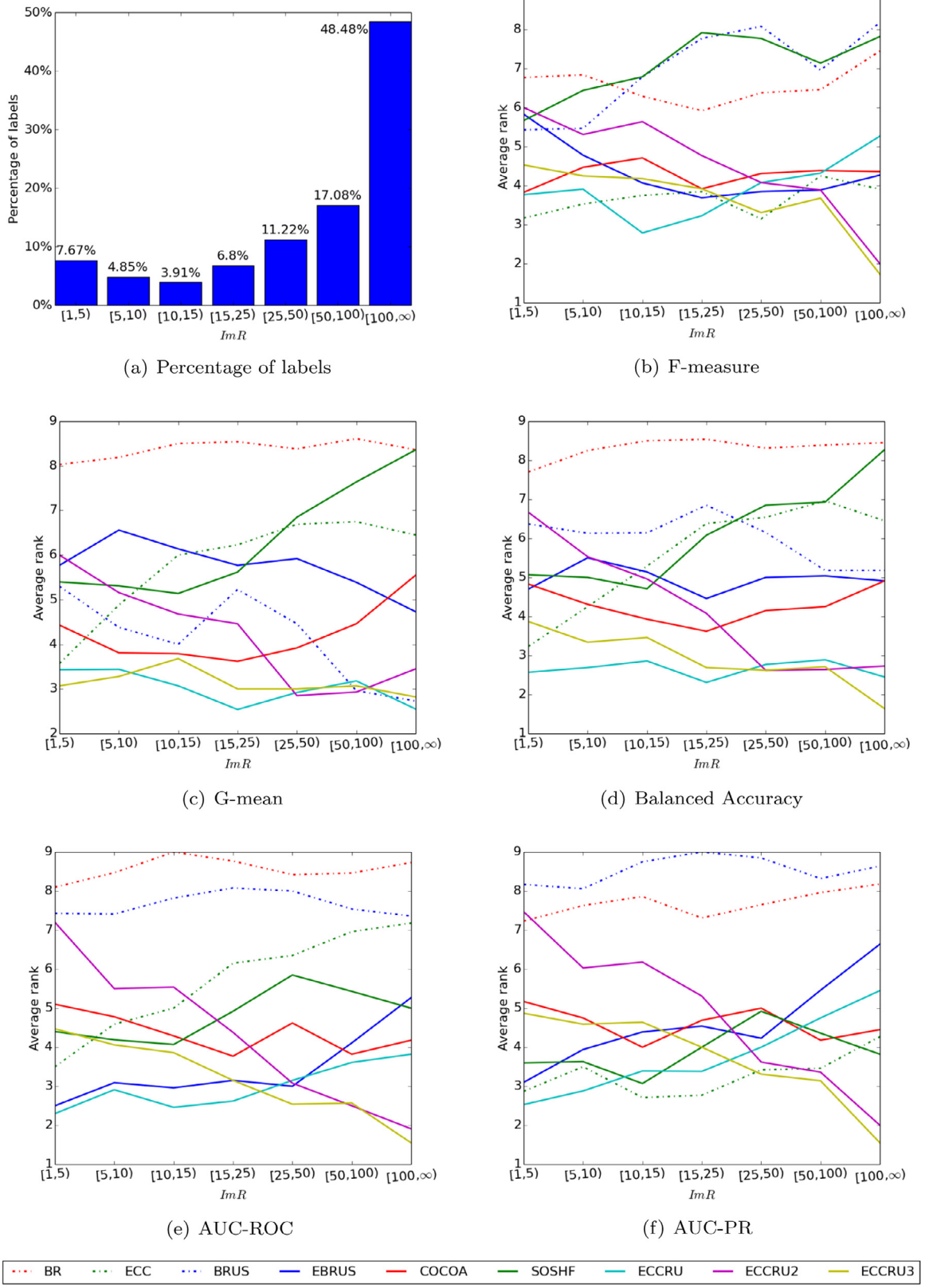
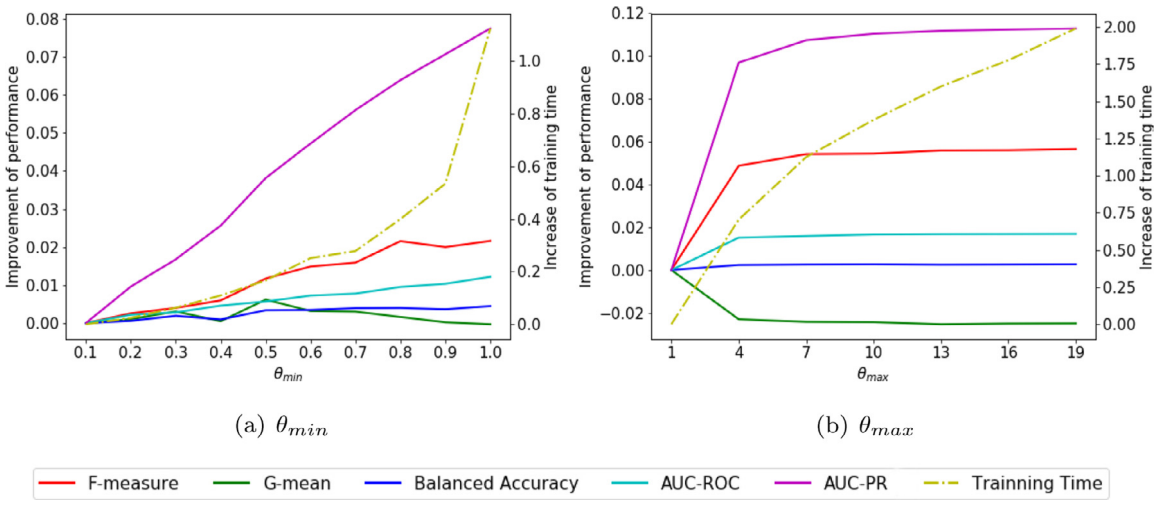


Fig. 4. Sub-figure (a) shows the percentage of labels and sub-figures (b)–(f) the average rank of all methods for different *ImR* intervals in terms of the 5 different evaluation metrics.



(a) θ_{min} (b) θ_{max}

Fig. 5. The increment of performance and training time of ECCRU3 with various parameters settings.

4.3. Accuracy under different imbalance levels

To investigate the accuracy of the competing methods under different imbalance levels, we divide *ImR* into 7 intervals: [1, 5), [5, 10), [10, 15), [15, 25), [25, 50), [50, 100), [100, ∞). Fig. 4(a) shows the percentages of the labels from all 16 datasets that fall into these intervals. We can see that nearly 2/3 of labels have $ImR \geq 50$ and only 16% of the labels have $ImR < 15$. Fig. 4(b)–(f) shows the average rank of the 9 competing methods based on the 5 evaluation metrics calculated on each subset of all labels belonging to each interval. We can see that the proposed approaches dominate the rankings for all measures when $ImR \geq 50$. When $ImR < 50$ ECC does well in F-measure AUC-PR.

We also notice that with the exception of Balanced Accuracy, ECCRU3 dominates ECCRU2 in all other measures for roughly all levels of *ImR*, and ECCRU dominates ECCRU2 and ECCRU3 for $ImR < 25$. We hypothesize that the observed behavior is due to the following reason: when imbalance is not that large, then it is more important to build the full chains of ECCRU, in order to gain from modeling the dependencies among the labels, similarly to ECC. When imbalance is large, then it is more important to direct training effort towards exploiting more of the majority samples in imbalanced labels that are part of smaller chains. In other words, given the same budget of training examples, when *ImR* is high, then the benefits from exploiting more majority training examples surpass the benefits of modeling label dependencies.

4.4. Analysis of the parameters of ECCRU3

Here, we investigate the influence that different values of the θ_{min} and θ_{max} parameters have on ECCRU3. Fig. 5(a) presents the relative change in accuracy and training time for $\theta_{min} \in \{0.1, 0.2, \ldots, 1.0\}$ compared to setting θ_{min} equal to 0.1. Formally, if p_z is the accuracy (left axis) or training time of ECCRU3 when $\theta_{min} = z$, then the vertical axis of Fig. 5(a) shows $(p_z - p_{0.1})/p_{0.1}$. The scale of the left axis concerns the accuracy measures, while the scale of the right axis concerns the training time. Apart from θ_{min}, other parameters of ECCRU3 are same with Section 4.1. As θ_{min} increases, more binary classifiers are built for balanced labels, leading to an increase in training time, but also in accuracy, with the exception of G-mean when $\theta_{min} > 0.5$. The relative increase of accuracy is small (largest one is about 0.08 in terms of AUC-PR) compared to the relative increase in training time (more than 1.0 when $\theta_{min} = 1.0$).

Similarly, Fig. 5(b) shows the relative change in the accuracy and training time of ECCRU3 for $\theta_{max} \in \{1, 4, 7, 10, 13, 16, 19\}$ compared to setting θ_{max} equal to 1. Except for θ_{max}, other parameters of ECCRU3 are same with Section 4.1. The change is steep from $\theta_{max} = 1$ to $\theta_{max} = 4$, small from $\theta_{max} = 4$ to $\theta_{max} = 10$ and negligible for $\theta_{max} > 10$ for all accuracy measures, with the exception of Balanced Accuracy. As θ_{max} increases, ECCRU3 performs better in terms of AUC-PR, F-measure and AUC-ROC, while it becomes worse in terms of G-mean. In terms of Balanced Accuracy, ECCRU3 is not sensitive to the variation of θ_{max}. As noticed in Fig. 5(a), the changes in accuracy are small compared to the increase in training time.

4.5. Inverse imbalanced label

Finally, we focus on what we call the inverse imbalanced label issue, the special case where the presence of a label (class "1") is the majority class. Although the majority class of most labels is "0", there are still some exceptions. Table 5 presents the 7 datasets that contain inverse imbalanced labels and were used in the experiments of this part, along with the percentage of inverse imbalanced labels and their average *ImR*, denoted as %IIL and $\overline{ImR}$ respectively.

Most evaluation metrics (e.g. F-measure, AUC-PR, AUC-ROC) are asymmetric with respect to the classes and focus on the positive class ("1"). As a consequence, dealing with the inverse imbalance, by e.g. undersampling the positive class, is going to lead to worse accuracy. Therefore, to deal with the inverse imbalance situation, we modified the three proposed variants of ECCRU by removing the undersampling strategy for the inverse imbalanced labels. However, quitting undersampling for inverse imbalanced labels is a double-edged sword. On the one hand, it is beneficial to the prediction of inverse imbalanced labels because the binary classifiers would emphasize the majority class ("1"). On the other hand, due to the elimination of undersampling, all instances are leveraged in training the classifiers and the augmented features are always in-sample predictions, which enlarges the difference between the training and prediction distributions of the added features corresponding to these labels, which impacts the rest of the labels. In general, the larger the percentage of inverse imbalanced labels, the better the accuracy of the modified models. The higher the *ImR* of inverse imbalanced label, the higher the negative influence on other labels.

The results of the modified methods whose performance are better than their corresponding original approach are listed in Table 5. To analyze the results, the 7 datasets are categorized into 5 groups according to their percentage of inverse imbalanced labels and the average *ImR* of such labels.

Table 5
The information of inverse imbalanced labels and results of modified ECCRUs, where I, II and III denote that the modified ECCRU, ECCRU2 and ECCRU3 respectively outperform the corresponding original method, and # is the number of improved modified methods for each dataset (last column) and each evaluation metric (last row). Abbreviations of metrics are the same as in Table 3.

Dataset	% IIL	$\overline{ImR}$	*F*	*G*	*B*	*A*	*P*	#
enron	3.8%	1.083	I,II,III	I,II	II	II	I,II,III	10
tmc2007–500	4.5%	1.302	I,II,III	I,III	I,III	II,III	I,II,III	12
flags	42.9%	2.605	I	I,II,III	I,II,III	I	I,II,III	11
cal500	4.6%	2.319	II	–	II	II,III	I,II	6
mediamill	2%	2.561	II,III	II	II	–	II,III	6
yeast	14.3%	2.966	II,III	I	–	–	–	3
yahoo-Business1	3.6%	6.521	–	–	–	–	–	0
#			12	9	8	6	13	

Table A.1
Experiment result in terms of macro-averaged F-measure.

	BR	ECC	BRUS	EBRUS	COCOA	SOSHF	ECCRU	ECCRU2	ECCRU3
bibtex	0.1073	0.15	0.0992	**0.1768**	0.1748	0.1259	0.1752	0.1731	0.1742
cal500	0.1809	0.1579	0.2068	0.1236	0.1579	**0.2243**	0.1486	0.1473	0.1515
corel5k	0.0285	0.0446	0.0366	0.0498	0.0475	0.0299	**0.0514**	0.0475	0.0488
enron	0.1416	0.1866	0.1497	0.1939	0.1956	0.1524	0.1878	0.1955	**0.2001**
eurlex-sm	0.2731	**0.3382**	0.0923	0.2906	0.2866	0.2063	0.2933	0.3299	0.3335
flags	0.647	0.6621	0.6579	0.6733	0.677	0.6661	0.6638	**0.6787**	**0.6787**
genbase	0.8587	**0.9579**	0.761	0.8269	0.8339	0.6487	0.9369	0.9334	0.9438
mediamill	0.1752	**0.2486**	0.1042	0.1959	0.1938	0.2317	0.1772	0.2251	0.2293
medical	0.3807	0.4123	0.3732	0.4149	0.4075	0.2567	0.4134	0.4113	**0.4193**
rcv1subset1	0.1836	0.2296	0.1347	0.2247	0.2391	0.195	0.2269	**0.2407**	0.2395
rcv1subset2	0.1669	0.2083	0.1239	0.2159	0.2227	0.189	0.2127	**0.2288**	0.2283
scene	0.6183	0.7207	0.592	0.6378	0.6704	**0.7285**	0.7071	0.7025	0.7025
tmc2007–500	0.5016	0.5498	0.3995	0.5165	0.5542	**0.56**	0.5282	0.542	0.5451
yahoo-Arts1	0.1542	0.1998	0.1586	0.2058	0.2082	0.1699	0.2019	0.2069	**0.2129**
yahoo-Business1	0.1613	0.2065	0.1162	0.1918	0.1961	0.1717	0.1928	0.2133	**0.2162**
yeast	0.3905	0.4158	0.4141	0.376	0.406	**0.458**	0.411	0.3976	0.3976

Table A.2
Experiment result in terms of macro-averaged G-mean.

	BR	ECC	BRUS	EBRUS	COCOA	SOSHF	ECCRU	ECCRU2	ECCRU3
bibtex	0.2134	0.4841	0.7159	0.6606	0.6749	0.4393	**0.7198**	0.7184	0.7174
cal500	0.1996	0.3299	**0.4793**	0.2643	0.3044	0.3663	0.3623	0.3121	0.3178
corel5k	0.0572	0.1291	0.3168	0.3435	**0.3728**	0.0918	0.3501	0.3284	0.3294
enron	0.1656	0.3909	**0.5657**	0.4984	0.493	0.3003	0.5467	0.5029	0.5173
eurlex-sm	0.385	0.5772	0.7238	0.68	0.6892	0.3806	**0.7452**	0.7157	0.7179
flags	0.4477	0.6468	0.6445	**0.6547**	0.6528	0.6098	0.6522	0.6541	0.6541
genbase	0.8719	**0.9719**	0.8573	0.8597	0.8433	0.7443	0.9639	0.9644	0.9647
mediamill	0.3251	0.5441	0.6908	0.6523	0.6709	0.484	0.6662	0.688	**0.6957**
medical	0.419	0.5252	0.6962	0.6635	0.6472	0.381	**0.7382**	0.6823	0.6889
rcv1subset1	0.3592	0.5536	**0.7076**	0.6471	0.6588	0.4955	0.6989	0.6864	0.6928
rcv1subset2	0.3545	0.5276	**0.7008**	0.6299	0.6516	0.5033	0.6911	0.6762	0.6833
scene	0.7493	0.8338	0.7917	0.7527	0.7835	0.8332	0.828	**0.8341**	**0.8341**
tmc2007–500	0.7187	0.7978	0.8191	0.8151	0.8328	0.8165	**0.8374**	0.836	0.8371
yahoo-Arts1	0.2718	0.4715	**0.5823**	0.5512	0.5412	0.4526	0.5737	0.5721	0.5787
yahoo-Business1	0.2571	0.4652	0.6055	0.5659	0.5717	0.4417	0.6193	0.6221	**0.6296**
yeast	0.4739	0.5172	0.5621	0.4366	0.4981	**0.567**	0.54	0.5151	0.5151

- For the *enron* and *tmc2007-500* datasets with lower percentage (<5%) and $\overline{ImR}$ (<1.5), the modified approaches are better than the original ones in most cases, due to the low $\overline{ImR}$.
- For the *flags* dataset with nearly half of inverse imbalanced labels and moderate $\overline{ImR}$ (around 2.5), the advantage of modified methods is obvious as well, mainly coming from the high percentage.
- For the *cal500* and *mediamill* datasets with low percentage and moderate $\overline{ImR}$, modified models outperform original ones in only 6 out of the 15 cases.
- For the *yeast* dataset with high percentage (nearly 15%) and high $\overline{ImR}$, the damage caused by high $\overline{ImR}$ is usually larger than the improvement introduced by the high percentage.
- For the *yahoo-Business1* dataset with low percentage and the highest $\overline{ImR}$, the latter factor makes the modified methods perform worse.

Furthermore, from the view of evaluation metrics, the superiority of modified models are more notable in terms of F-measure and AUC-PR than other three metrics. F-measure and AUC-PR emphasize on positive ("1") class, while G-mean, Balanced Accuracy and AUC-ROC just treat the two class equally. Without the undersampling (within bias to class "1") for inverse imbalanced labels, improved performance of those labels is more remarkable in terms of F-measure and AUC-PR than other three metrics.

5. Conclusions and future work

We started from a strong and theoretically grounded multi-label learning algorithm, ECC, and made it resilient to the challenge of class imbalance by employing random undersampling to balance the class distribution of each binary training set, leading to the ECCRU algorithm. We then discussed approaches to make the best exploitation of a computational budget based on the key observation that different imbalance ratios lead to different

Table A.3
Experiment result in terms of macro-averaged Balanced Accuracy.

	BR	ECC	BRUS	EBRUS	COCOA	SOSHF	ECCRU	ECCRU2	ECCRU3
bibtex	0.5716	0.6299	0.7194	0.7138	0.724	0.6232	**0.7461**	0.746	0.7458
cal500	0.5085	0.5157	0.5144	0.5136	0.5166	0.5141	**0.5217**	0.5167	0.5188
corel5k	0.5113	0.5249	0.5549	0.5661	**0.5731**	0.5172	0.5697	0.5665	0.5682
enron	0.5474	0.5981	0.6037	0.6217	0.6175	0.5667	**0.6301**	0.6225	0.63
eurlex-sm	0.6272	0.714	0.7591	0.7631	0.7711	0.6361	**0.7966**	0.7874	0.7907
flags	0.6144	0.6602	0.6584	**0.6689**	0.6642	0.6166	0.6674	0.6663	0.6663
genbase	0.9345	**0.9845**	0.9211	0.9236	0.9157	0.862	0.9756	0.9761	0.9765
mediamill	0.5761	0.6548	0.6939	0.7025	0.7102	0.6473	0.706	0.7228	**0.729**
medical	0.6907	0.7246	0.7909	0.7888	0.7841	0.6553	**0.8169**	0.7976	0.8021
rcv1subset1	0.592	0.6728	0.7288	0.7225	0.7314	0.668	0.7502	0.7471	**0.7534**
rcv1subset2	0.587	0.6598	0.7163	0.7102	0.7301	0.6646	0.7421	0.7433	**0.747**
scene	0.7666	0.8399	0.7922	0.7818	0.7994	**0.8428**	0.8354	0.839	0.839
tmc2007–500	0.7506	0.8089	0.8196	0.8214	0.837	0.8225	**0.8399**	0.8378	0.8395
yahoo-Arts1	0.553	0.5941	0.5929	0.6246	0.6182	0.5789	0.6288	0.6313	**0.635**
yahoo-Business1	0.5557	0.6152	0.6253	0.6526	0.6532	0.6153	0.6706	0.6774	0.6828
yeast	0.5682	0.593	0.571	0.575	0.5909	**0.6013**	0.597	0.5898	0.5898

Table A.4
Experiment result in terms of macro-averaged AUC-ROC.

	BR	ECC	BRUS	EBRUS	COCOA	SOSHF	ECCRU	ECCRU2	ECCRU3
bibtex	0.5734	0.7469	0.7218	0.8301	**0.8399**	0.8046	0.8324	0.8362	0.8361
cal500	0.5041	0.5223	0.514	0.5346	0.5339	0.537	**0.5385**	0.5346	0.5374
corel5k	0.5118	0.5309	0.5554	0.5989	0.6036	0.6001	0.6305	0.6346	**0.639**
enron	0.5442	0.6351	0.603	0.6786	0.6721	0.6675	0.6848	0.6761	**0.6898**
eurlex-sm	0.6406	0.774	0.7568	0.8841	0.8795	0.8591	0.8833	0.8964	**0.9031**
flags	0.6267	**0.7348**	0.6639	0.7238	0.7179	0.7015	0.7319	0.7314	0.7314
genbase	0.9346	**0.9979**	0.9211	0.9269	0.931	0.9629	0.9811	0.99	0.99
mediamill	0.5962	0.7523	0.692	0.8237	0.8112	0.8221	0.81	0.8192	**0.8302**
medical	0.6918	0.7677	0.7926	0.85	0.8572	0.8566	0.8788	0.898	**0.9014**
rcv1subset1	0.5916	0.7388	0.7312	0.8654	0.8546	0.8347	0.8691	0.8763	**0.884**
rcv1subset2	0.5905	0.7365	0.7201	0.8506	0.8458	0.8119	0.8541	0.8582	**0.865**
scene	0.7561	0.9196	0.7883	0.9235	0.9153	**0.9243**	0.9222	0.9214	0.9214
tmc2007–500	0.7321	0.8915	0.8169	0.9027	**0.9113**	0.8952	0.9079	0.9051	0.9089
yahoo-Arts1	0.5451	0.6318	0.5996	0.6886	0.6801	0.6341	0.6811	0.687	**0.6897**
yahoo-Business1	0.5459	0.6438	0.6313	0.7417	0.7315	0.7125	0.7396	0.7439	**0.7507**
yeast	0.5679	0.6494	0.5733	0.6576	0.6556	**0.6624**	0.6584	0.6513	0.6513

Table A.5
Experiment result in terms of macro-averaged AUC-PR.

	BR	ECC	BRUS	EBRUS	COCOA	SOSHF	ECCRU	ECCRU2	ECCRU3
bibtex	0.0733	0.1286	0.0529	0.1218	**0.1383**	0.1189	0.1287	0.1275	0.1294
cal500	0.1561	0.1812	0.1569	0.1818	0.1838	**0.1863**	0.1847	0.1813	0.1847
corel5k	0.0194	0.0361	0.016	0.0357	0.0349	0.0334	**0.0429**	0.036	0.0401
enron	0.1111	0.1813	0.1028	0.1716	0.1731	0.1742	0.1752	0.1773	**0.1892**
eurlex-sm	0.1763	**0.327**	0.0528	0.2339	0.2418	0.26	0.2515	0.2938	0.3126
flags	0.5999	0.7025	0.6185	0.6895	0.6957	0.6843	**0.7066**	0.7053	0.7053
genbase	0.8471	**0.9679**	0.7346	0.8108	0.8429	0.7631	0.9317	0.947	0.9497
mediamill	0.1105	0.2219	0.0659	0.1529	0.1444	**0.2411**	0.1479	0.1861	0.2001
medical	0.3368	**0.4176**	0.3108	0.3735	0.3832	0.3642	0.3797	0.4004	0.4115
rcv1subset1	0.1118	0.2088	0.0731	0.1927	0.199	0.2066	0.2042	0.2106	**0.2228**
rcv1subset2	0.1003	0.1873	0.0661	0.1804	0.1844	0.1942	0.1869	0.195	**0.2054**
scene	0.5091	0.7874	0.4322	0.7679	0.7503	**0.7935**	0.7718	0.769	0.769
tmc2007–500	0.385	0.5431	0.265	0.4657	0.5219	**0.56**	0.4867	0.5062	0.515
yahoo-Arts1	0.1125	0.1846	0.0939	0.1876	0.1847	0.167	0.1838	0.1805	**0.1889**
yahoo-Business1	0.1068	0.1723	0.0759	0.154	0.1564	**0.1849**	0.1663	0.1733	0.182
yeast	0.3544	0.452	0.3432	0.453	0.4529	**0.4554**	0.449	0.4394	0.4394

levels of exploitation of majority examples, leading to the ECCRU2 and ECCRU3 algorithms. Our empirical study showed that the proposed methods are competitive to related benchmark and state-of-the-art methods, and especially ECCRU3 achieves the best performance in terms of five imbalance metrics with positive significance tests in almost all comparisons.

We also presented an interesting analysis of the behavior of the algorithms under different levels of class imbalance, and discussed insights on the causes of this behavior. Finally, we investigated the special case where the positive, instead of the negative, class is associated with the majority of the examples and analyzed the sensitivity of the proposed methods in terms of different parameter settings.

An interesting future research direction is to address class imbalance in the context of active multi-label learning [38], especially in situations where the budget for data annotation is limited [39], where having multiple imbalanced labels increases the complexity of the problem.

Acknowledgment

Bin Liu is supported from the China Scholarship Council (CSC) under the Grant CSC No. 201708500095.

Appendix

See Tables A.1–A.6.

Table A.6
Training time (in second) of comparison and proposed methods.

	BR	ECC	BRUS	EBRUS	COCOA	SOSHF	ECCRU	ECCRU2	ECCRU3
bibtex	87.1	1757	**22**	253	225	848	300	278	282
cal500	2.444	33.4	**1.616**	20.2	10.8	86.9	14.7	14.8	15.6
corel5k	195	1748	**89.6**	905	265	856	605	750	953
enron	2.571	37.9	**1.161**	11.4	7.994	151	15.1	34.2	37.3
eurlex-sm	3651	71 171	**223**	2145	1746	1526	2834	7670	7814
flags	0.2704	0.5398	**0.0194**	0.1794	0.0678	38.5	0.1729	0.1623	0.1081
genbase	0.78	5.836	**0.6326**	3.82	3.324	278	3.062	14.8	10.3
mediamill	825	13 869	**106**	1201	543	4079	1558	1457	1695
medical	0.5254	5.729	**0.2603**	2.257	2.238	74.3	2.716	10.2	9.997
rcv1subset1	435	4940	**45.3**	495	427	1475	603	651	739
rcv1subset2	462	4735	**40.8**	482	390	1325	572	603	684
scene	3.476	25.2	**0.8777**	11.9	11.7	414	9.948	8.071	6.935
tmc2007–500	3145	44 400	**2098**	21 473	6058	7499	28 169	6777	13 757
yahoo-Arts1	162	1444	**16.8**	167	144	605	191	176	201
yahoo-Business1	219	1780	**16.6**	167	90.8	638	199	206	257
yeast	2.819	23	**1.403**	19.7	18.1	563	16.6	10.2	10.1

References

[1] T.N. Rubin, A. Chambers, P. Smyth, M. Steyvers, Statistical topic models for multi-label document classification, Mach. Learn. 88 (1–2) (2012) 157–208, http://dx.doi.org/10.1007/s10994-011-5272-5.

[2] J. Read, B. Pfahringer, G. Holmes, E. Frank, Classifier chains for multi-label classification, Mach. Learn. 85 (3) (2011) 333–359, http://dx.doi.org/10.1007/s10994-011-5256-5.

[3] K. Dembczyński, W. Cheng, E. Hüllermeier, Bayes optimal multilabel classification via probabilistic classifier chains, in: Proceedings of the 27th International Conference on Machine Learning (ICML-10), 2010, pp. 279–286.

[4] L. Breiman, J.H. Friedman, R.A. Olshen, C.J. Stone, Classification and Regression Trees, Chapman and Hall/CRC, 1984, p. 368, http://dx.doi.org/10.1371/journal.pone.0015807.

[5] B. Liu, G. Tsoumakas, Making classifier chains resilient to class imbalance, in: 10th Asian Conference on Machine Learning (ACML 2018), Beijing, 2018, pp. 280–295.

[6] Z.A. Daniels, D.N. Metaxas, Addressing imbalance in multi-label classification using structured hellinger forests, in: Proceedings of the 31st AAAI Conference on Artificial Intelligence, 2017, pp. 1826–1832.

[7] E. Spyromitros-Xioufis, G. Tsoumakas, W. Groves, I. Vlahavas, Multi-target regression via input space expansion: treating targets as inputs, Mach. Learn. 104 (1) (2016) 55–98, http://dx.doi.org/10.1007/s10994-016-5546-z.

[8] H. He, E.A. Garcia, Learning from imbalanced data, IEEE Trans. Knowl. Data Eng. 21 (9) (2009) 1263–1284, http://dx.doi.org/10.1109/TKDE.2008.239.

[9] Y. Yan, M. Tan, Y. Xu, J. Cao, M. Ng, H. Min, Q. Wu, Oversampling for imbalanced data via optimal transport, in: Proceedings of the AAAI Conference on Artificial Intelligence, 2019, pp. 5605–5612.

[10] Q. Wu, Y. Ye, H. Zhang, M.K. Ng, S.-S. Ho, ForesTexter: an efficient random forest algorithm for imbalanced text categorization, Knowl.-Based Syst. 67 (2014) 105–116.

[11] C.X. Ling, V.S. Sheng, Q. Yang, Test strategies for cost-sensitive decision trees, IEEE Trans. Knowl. Data Eng. 18 (8) (2006) 1055–1067.

[12] M. Galar, A. Fernandez, E. Barrenechea, H. Bustince, F. Herrera, A review on ensembles for the class imbalance problem: Bagging-, boosting-, and hybrid-based approaches, IEEE Trans. Syst. Man Cybernet. C (Appl. Rev.) 42 (4) (2012) 463–484, http://dx.doi.org/10.1109/tsmcc.2011.2161285.

[13] F. Charte, A. Rivera, M.J. del Jesus, F. Herrera, A first approach to deal with imbalance in multi-label datasets, in: Proceedings of the 8th International Conference on Hybrid Artificial Intelligent Systems (HAIS 2013), in: LNAI, vol. 8073, 2013, pp. 150–160, http://dx.doi.org/10.1007/978-3-642-40846-5_16.

[14] F. Charte, A.J. Rivera, M.J. del Jesus, F. Herrera, Addressing imbalance in multilabel classification: Measures and random resampling algorithms, Neurocomputing 163 (2015) 3–16, http://dx.doi.org/10.1016/j.neucom.2014.08.091.

[15] F. Charte, A.J. Rivera, M.J. Del Jesus, F. Herrera, MLeNN: A first approach to heuristic multilabel undersampling, in: Intelligent Data Engineering and Automated Learning – IDEAL 2014, in: LNCS, vol. 8669, Springer International Publishing, 2014, pp. 1–9, http://dx.doi.org/10.1007/978-3-319-10840-7_1.

[16] F. Charte, A.J. Rivera, M.J. Del Jesus, F. Herrera, MLSMOTE: Approaching imbalanced multilabel learning through synthetic instance generation, Knowl.-Based Syst. 89 (2015) 385–397, http://dx.doi.org/10.1016/j.knosys.2015.07.019.

[17] F. Charte, A. Rivera, M.J. Del Jesus, F. Herrera, Resampling multilabel datasets by decoupling highly imbalanced labels, in: Lecture Notes in Artificial Intelligence (Subseries of Lecture Notes in Computer Science), vol. 9121, 2015, pp. 489–501, http://dx.doi.org/10.1007/978-3-319-19644-2_41.

[18] M.R. Boutell, J. Luo, X. Shen, C.M. Brown, Learning multi-label scene classification, Pattern Recognit. 37 (9) (2004) 1757–1771, http://dx.doi.org/10.1016/j.patcog.2004.03.009.

[19] K. Chen, B.-L. Lu, J.T. Kwok, Efficient classification of multi-label and imbalanced data using min-max modular classifiers, in: The 2006 IEEE International Joint Conference on Neural Network Proceedings, IEEE, 2006, pp. 1770–1775, http://dx.doi.org/10.1109/IJCNN.2006.246893.

[20] S. Dendamrongvit, M. Kubat, Undersampling approach for imbalanced training sets and induction from multi-label text-categorization domains, in: Proceedings of the 13th Pacific-Asia International Conference on Knowledge Discovery and Data Mining (PAKDD'09), 2009, pp. 40–52, http://dx.doi.org/10.1007/978-3-642-14640-4_4.

[21] M.A. Tahir, J. Kittler, F. Yan, Inverse random under sampling for class imbalance problem and its application to multi-label classification, Pattern Recognit. 45 (10) (2012) 3738–3750, http://dx.doi.org/10.1016/j.patcog.2012.03.014.

[22] S. Wan, Y. Duan, Q. Zou, HPSLPred: An ensemble multi-label classifier for human protein subcellular location prediction with imbalanced source, Proteomics 17 (17–18) (2017) 1700262, http://dx.doi.org/10.1002/pmic.201700262.

[23] M.-L. Zhang, Y.-K. Li, X.-Y. Liu, Towards class-imbalance aware multi-label learning, in: Proceedings of the 24th International Conference on Artificial Intelligence, 2015, pp. 4041–4047.

[24] G. Tepvorachai, C. Papachristou, Multi-label imbalanced data enrichment process in neural net classifier training, in: Proceedings of the International Joint Conference on Neural Networks, 2008, pp. 1301–1307, http://dx.doi.org/10.1109/IJCNN.2008.4633966.

[25] C. Li, G. Shi, Improvement of learning algorithm for the multi-instance multi-label RBF neural networks trained with imbalanced samples, J. Inf. Sci. Eng. 29 (4) (2013) 765–776.

[26] K. Sozykin, A.M. Khan, S. Protasov, R. Hussain, Multi-label class-imbalanced action recognition in hockey videos via 3D convolutional neural networks, in: 19th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD), 2018, pp. 146–151.

[27] P. Cao, X. Liu, D. Zhao, O. Zaiane, Cost sensitive ranking support vector machine for multi-label data learning, in: Proceedings of the 16th International Conference on Hybrid Intelligent Systems (HIS 2016), Springer International Publishing, Cham, 2017, pp. 244–255.

[28] K.W. Sun, C.H. Lee, Addressing class-imbalance in multi-label learning via two-stage multi-label hypernetwork, Neurocomputing 266 (2017) 375–389, http://dx.doi.org/10.1016/j.neucom.2017.05.049.

[29] L. Li, H. Wang, Towards label imbalance in multi-label classification with many labels, 2016, arXiv preprint arXiv:1604.01304.

[30] B. Wu, S. Lyu, B. Ghanem, Constrained submodular minimization for missing labels and class imbalance in multi-label learning, in: Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, in: AAAI'16, AAAI Press, 2016, pp. 2229–2236.

[31] W. Zeng, X. Chen, H. Cheng, Pseudo labels for imbalanced multi-label learning, in: 2014 International Conference on Data Science and Advanced Analytics (DSAA), 2014, pp. 25–31, http://dx.doi.org/10.1109/DSAA.2014.7058047.

[32] G. Tsoumakas, E. Spyromitros-Xioufis, J. Vilcek, I. Vlahavas, MULAN: A java library for multi-label learning, J. Mach. Learn. Res. (2011).

[33] J.S. Akosa, Predictive accuracy : A misleading performance measure for highly imbalanced data classified negative, SAS Global Forum (2017).
[34] L. Tang, S. Rajan, V.K. Narayanan, Large scale multi-label classification via metalabeler, in: Proceedings of the 18th International Conference on World Wide Web - WWW '09, 2009, p. 211, http://dx.doi.org/10.1145/1526709.1526738.
[35] K. Sechidis, G. Tsoumakas, I. Vlahavas, On the stratification of multi-label data, in: Proc. 2011 European Conference on Machine Learning and Knowledge Discovery in Databases, Springer Berlin Heidelberg, Athens, Greece, 2011, pp. 145–158.
[36] S. Garcia, F. Herrera, An extension on "statistical comparisons of classifiers over multiple data sets" for all pairwise comparisons, J. Mach. Learn. Res. 9 (2008) 2677–2694.
[37] A. Benavoli, G. Corani, F. Mangili, Should we really use post-hoc tests based on mean-ranks? J. Mach. Learn. Res. 17 (2016) 1–10.
[38] E.A. Cherman, Y. Papanikolaou, G. Tsoumakas, M.C. Monard, Multi-label active learning: key issues and a novel query strategy, Evol. Syst. 10 (1) (2019) 63–78.
[39] Y. Zhang, P. Zhao, J. Cao, W. Ma, J. Huang, Q. Wu, M. Tan, Online adaptive asymmetric active learning for budgeted imbalanced data, in: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, ACM, 2018, pp. 2768–2777.