Making Classifier Chains Resilient to Class Imbalance

Bin Liu binliu@csd.auth.gr
Grigorios Tsoumakas greg@csd.auth.gr
School of Informatics, Aristotle University of Thessaloniki, Thessaloniki, Greece

Abstract

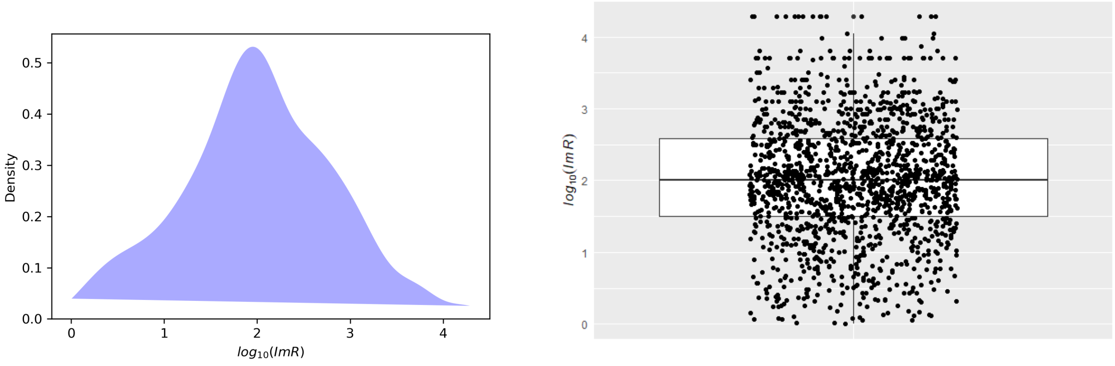
Class imbalance is an intrinsic characteristic of multi-label data. Most of the labels in multi-label data sets are associated with a small number of training examples, much smaller compared to the size of the data set. Class imbalance poses a key challenge that plagues most multi-label learning methods. 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.

Keywords: Multi-label learning, class imbalance, classifier chains, undersampling

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 (Rubin et al., 2012) — 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 examples of the majority class over the number of examples of the minority class. Figure 1 (a) shows a density estimation plot and Figure 1 (b) a box-plot of the imbalance ratios of 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) (Read et al., 2011), a popular multi-label learning algorithm with state-of-the-art predictive performance that is also accompanied by a theoretical interpretation based on probability theory (Dembczyński et al., 2010). 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.



(a) (b)

Figure 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.

To make ECC resilient to class imbalance we contribute a new approach that couples it with random undersampling (Breiman et al., 1984). We then present two extensions of this basic approach, in order to improve the exploitation of majority 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.

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, $L = \{l_1, l_2, ..., l_q\}$ 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 j-th element of $\boldsymbol{y}_i$ and $y_{ij} = 1(0)$ denotes that l_j is (not) associated with i-th 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 real-valued vector $\hat{\boldsymbol{y}}$ with the predicted labels of or 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 (Read et al., 2011). 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 j-th 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 $\{h_1, ..., h_q\}$ 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 (Read et al., 2011). 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 a voting strategy. The j-th 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 (Breiman et al., 1984), in order to balance the class distribution of each binary training set. This leads to the classifier chain with random undersampling approach (CCRU), whose pseudocode is shown in Algorithm 1.

CCRU builds binary classifiers sequentially according to label sequence CH. Random undersampling of majority examples is applied to each binary training set before building the corresponding classifier (line 4). In specific, $M_j - m_j$ majority class examples are randomly removed from each label l_j in order to create a fully balanced 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 (Spyromitros-Xioufis et al., 2016): 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 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 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 and the equal number of retained majority examples, and out-of-sample for the rest of the majority examples that were removed (lines 7-14).

Algorithm 1: Training of CCRU

```
input  : multi-label data set: $D$, sequence of labels: $CH$
output: CCRU model: $h = \{h_1, ..., h_{|CH|}\}$
$D_1 \leftarrow \{(\boldsymbol{x}_1, y_{1CH_1}), ..., (\boldsymbol{x}_{|D|}, y_{|D|CH_1})\}$ ;
$h \leftarrow \emptyset$ ;
for $j \leftarrow 1$ to $|CH|$ do
    $D_j^* \leftarrow$ RandomUnderSample($D_j$) ;        /* apply random undersampling to $D_j$ */
    train $h_j$ based on $D_j^*$ ;
    $h \leftarrow h \cup h_j$ ;
    if $j < |CH|$ then
        $D_{j+1} \leftarrow \emptyset$ ;
        foreach $(\boldsymbol{x}, y)$ in $D_j$ do
            $\hat{y}_{CH_j} \leftarrow h_j(\boldsymbol{x})$ ;
            $\boldsymbol{x}' \leftarrow [x_1, ..., x_d, \hat{y}_{CH_1}, ..., \hat{y}_{CH_j}]$ ;        /* add augmented features */
            $D_{j+1} \leftarrow D_{j+1} \cup (\boldsymbol{x}', y_{CH_{j+1}})$ ;
        end
    end
end
return $h = \{h_1, ..., h_{|CH|}\}$ ;
```

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.

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 Equation 1.

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

Figure 2 plots Equation 1 for 10 chains, 1,000 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.

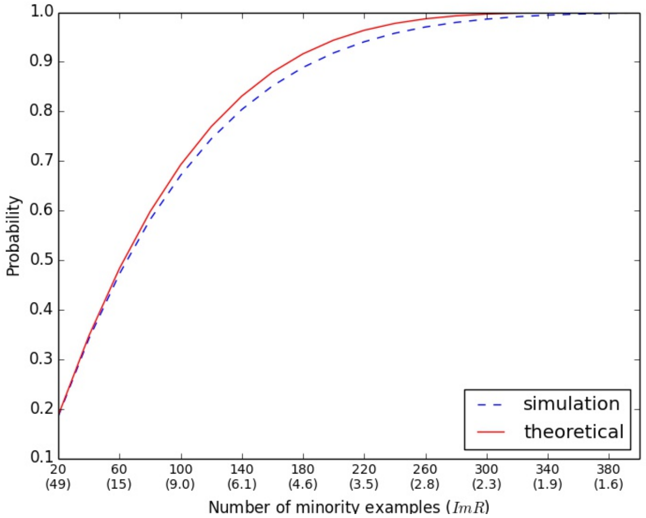


Figure 2: The empirically estimated (simulated) and theoretically approximated probability of a majority example of a label being retained for training by at least one of the 10 corresponding models of ECCRU with 10 chains, assuming 1,000 training examples and a number of minority examples varying from 20 to 400 with a step of 20. For the simulation, the sampling process was conducted 10,000 times.

A straightforward way to increase the exploitation of majority examples when ImR is high is to increase the number of chains. This is also theoretically grounded based on Equation 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. Consider for example a dataset with 100 training examples and 3 labels, each with 10, 20 and 30 minority examples respectively. The classifier of the first, second and third label will be trained with 20, 40 and 60 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 600 (10 times 10+20+30). Our approach divides this computational budget equally across labels, i.e. 200 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. 20, 10 and $6.\overline{6}$. 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 Equation 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 an increasingly smaller subset of labels until the minimum size of two labels. Continuing the aforementioned example, ECCRU2 would build 6 chains containing all three labels and 4 chains including the first two labels.

The pseudo-code of the training process of ECCRU2 is given in Algorithm 2. Firstly, the number of classifiers trained for each label c_j is calculated according to Equation (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 testing process of ECCRU2 is given in Algorithm 3. 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 $j = 1, ...q$. The rest parts of testing process of ECCRU2 are as in ECC.

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 testing process of ECCRU3 are the same with ECCRU2.

2.5. Complexity Analysis

Let's define $\Theta_{tr}(m_j, d)$ and $\Theta_{te}(d)$ the complexity of training and testing a binary classifier for label l_j, respectively. The complexity of ECCRU is $O\left(c \sum_{j=1}^{q} \Theta_{tr}(m_j, d) + ncq\Theta_{te}(d)\right)$ for training and $O\left(cq\Theta_{te}(d)\right)$ for testing. The training and testing complexity of ECCRU2 is $O\left(\frac{c}{q} \sum_{k=1}^{q} m_k * \sum_{j=1}^{q} \left(\frac{1}{m_j}\Theta_{tr}(m_j, d)\right) + n\Theta_{te}(d) \sum_{j=1}^{q} c_j\right)$ and $O\left(\Theta_{te}(d) \sum_{j=1}^{q} c_j\right)$. For both algorithms, the first part of the training complexity concerns building classifiers and the second relates to generating the augmented feature space. The classifiers in ECCRU2 are more than in ECCRU, which results in larger testing complexity and large complexity of generating augmented features. However, the comparison between the training complexity of the first part of ECCRU2 and ECCRU depends on the m_j and $\Theta_{tr}(m_j, d)$ of each label. The formulation of the training and testing complexity of ECCRU3 is the same with ECCUR2, 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.

Algorithm 2: 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: $\theta_{max}$
output: ECCRU2 model: $h = \{h^1, ..., h^{c'}\}$
for $j \leftarrow 1$ to $q$ do
    calculate $c_j$ according to (2) ;
    $c_j \leftarrow \min\{c_j, c\theta_{max}\}$;
    $cn_j \leftarrow c_j$ ;        /* the number of classifiers needed to be built for each label */
end
$c' \leftarrow 0$ ;                /* the counter to record the number of chains built actually */
for $i \leftarrow 1$ to $c\theta_{max}$ do
    $S \leftarrow \emptyset$ ;
    for $j \leftarrow 1$ to $q$ do
        if $cn_j > 0$ then
            $S \leftarrow S \cup j$ ;
            $cn_j \leftarrow cn_j - 1$ ;
        end
    end
    if $|S| < 2$ then
        break;
    end
    $CH^i \leftarrow$ RandomPermute($S$) ;         /* generate a chain by random permutation */
    $D' \leftarrow$ SampleWithReplacement($D$) ;         /* sample the $D$ with replacement */
    $h^i \leftarrow$ TrainCCRU($D'$, $CH^i$) ;         /* train a CCRU according to Algorithm 1 */
    $c' \leftarrow c' + 1$ ;
end
$h \leftarrow \{h^1, ..., h^{c'}\}$ ;
return $h = \{h^1, ..., h^{c'}\}$ ;
```

3. Related Work

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 (Charte et al., 2013). ML-RUS and ML-ROS delete instances with majority labels and clone examples with minority labels, respectively (Charte et al., 2015b). 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 (Charte et al., 2014). 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 (Charte et al., 2015c). REMEDIAL decomposes each complex instance into two easier instances, one of which merely contains majority labels and another only with minority labels (Charte et al., 2015a).

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,

Algorithm 3: Testing of ECCRU2

```
input  : test instance $\boldsymbol{x}$, number of labels: $q$, ECCRU2 model: $h = \{h^1, h^2, ..., h^{c'}\}$
output: relevance degree 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 $h^i_j$ ;
        $cc_k \leftarrow cc_k + 1$ ;
        $\boldsymbol{x}' \leftarrow [x_1, ..., x_d, h^i_1(x), ..., h^i_{j-1}(x)]$ ;
        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 $\hat{\boldsymbol{y}}$ ;
```

as BR does (Boutell et al., 2004), and using sampling or an ensemble strategy to solve each imbalanced binary classification problem (Chen et al., 2006; Dendamrongvit and Kubat, 2010; Tahir et al., 2012; Wan et al., 2017). Cross-Coupling Aggregation (COCOA) (Zhang et al., 2015) 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. The Sparse Oblique Structured Hellinger Forests (SOSHF) (Daniels and Metaxas, 2017) 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, such as neural network based (Tepvorachai and Papachristou, 2008; Li and Shi, 2013; Sozykin et al., 2017), SVM based (Cao et al., 2017) and hypernetwork based (Sun and Lee, 2017). Finally, other strategies, such as representation learning (Li and Wang, 2016), constrained submodular minimization (Wu et al., 2016) and balanced pseudo-label (Zeng et al., 2014), have been utilized to address the imbalance problem of multi-label learning as well.

Compared to the above approaches, the strengths of the proposed methods 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 (Zhang et al., 2015) that is second-order and (Chen et al., 2006; Dendamrongvit and Kubat, 2010; Tahir et al., 2012; Wan et al., 2017) that are first-order methods. Thirdly, it is algorithm independent, as it can be combined with any binary classifier that best fits the problem at hand, in contrast to (Tepvorachai and Papachristou, 2008; Li and Shi, 2013; Sozykin et al., 2017; Cao et al., 2017; Sun and Lee, 2017; Daniels and Metaxas, 2017) that build on top of particular learning paradigms.

4. Empirical Analysis

We first introduce the setup of our experiments. Then we present the experimental results and their analysis.

4.1. Setup

Our empirical study is based on 16 multi-label data sets obtained from Mulan's GitHub repository[1] (Tsoumakas et al., 2011). 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 (Zhang et al., 2015).

The proposed approaches are compared against five multi-label learning methods. Two of them are imbalance agnostic ones, namely the Binary Relevance (BR) baseline (Boutell et al., 2004) and the state-of-the-art ECC (Read et al., 2011), on which the proposed approaches build on. The other three methods 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 (Zhang et al., 2015). In ECCRU2 and ECCRU3, the θ_{max} is set to 10. The θ_{min} in ECCRU3 is set to 0.5. The ensemble size is set to 10 for all ensemble methods (ECC, EBRUS, ECCRU*). For COCOA in particular, the number of coupling class labels is set to $min(q-1, 10)$ as in (Zhang et al., 2015). A decision tree is used as the base classifier in all methods.

We employ five widely used binary metrics for imbalanced data (He and Garcia, 2009; Akosa, 2017): 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 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 $t \in \{0, 0.05, \dots, 1\}$ 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 (Tang et al., 2009). We apply 5 × 2-fold cross validation with multi-label stratification (Sechidis et al., 2011) to each dataset and the average results are reported.

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

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	$MeanImR$	$MaxImR$	$CVImR$
bibtex	text	7395	183	159	2.402	87.699	144	0.410
cal500	music	502	68	174	26.044	22.345	99.4	1.129
corel5k	image	5000	499	374	3.522	845.284	4999	1.528
enron	text	1702	100	53	3.378	136.867	1701	1.974
eurlex-sm	text	19348	500	201	2.213	2420.775	19347	2.136
flags	image	194	19	7	3.392	2.753	6.462	0.711
genbase	biology	662	1186	27	1.252	143.458	661	1.460
mediamill	video	43907	120	101	4.376	331.439	1415.355	1.178
medical	text	978	144	45	1.245	328.069	977	1.151
rcv1subset1	text	6000	472	101	2.88	235.58	2999	2.089
rcv1subset2	text	6000	472	101	2.634	190.906	1999	1.724
scene	image	2407	144	6	1.074	4.662	5.613	0.148
tmc2007-500	text	28596	500	22	2.158	25.823	63.844	0.791
yahoo-Arts1	text	7484	231	26	1.548	384.756	7483	3.816
yahoo-Business1	text	11214	219	30	1.437	1014.363	11213	2.813
yeast	biology	2417	103	14	4.237	8.954	70.088	1.997

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

	BR	ECC	BRUS	EBRUS	COCOA	ECCRU	ECCRU2	ECCRU3
F-measure	6.44	3.09	6.56	5.31	4.22	4.34	3.22	**2.81**
G-mean	7.75	5.81	3.31	5.63	5.00	**2.50**	3.28	2.72
Balanced Acc.	7.63	5.38	6.25	5.38	4.50	2.38	2.59	**1.91**
AUC-ROC	7.81	5.50	6.88	3.81	4.31	3.00	2.84	**1.84**
AUC-PR	7.00	2.63	7.88	5.06	4.50	3.28	3.59	**2.06**
Training Time	2.88	7.69	**1.00**	5.06	3.19	5.38	5.00	5.81

We obtained the code of the five existing methods from Mulan. Our approaches were also implemented in the context of Mulan. The experiments were conducted on a machine with 4 10-core CPUs running at 2.27 GHz.

4.2. Results and Analysis

We discuss the experimental results from two aspects. We first report the accuracy and training time of all participating methods over the 16 multi-label datasets and present results of significance tests. We then discuss how different methods behave under different levels of imbalance ratio.

Table 2 shows the average rank of each method in terms of the five evaluation metrics plus the training time, with the best result highlighted with bold typeface. 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 second position. In terms of training time, BRUS achieves the best results followed by BR and COCOA. The proposed methods come next, followed by ECC that achieves the worst results. To examine the statistical significance of the differences between the different 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 (Garcia and Herrera, 2008; Benavoli et al., 2016). Table 3 presents the results. We notice that ECCRU3 achieves the most significant wins than any other method in all measures: 4 in F-measure, 4 in G-mean together with BRUS and the other two variations of the proposed approach, 6 in Balanced Accuracy, 7 in AUC-ROC, and 6 in AUC-PR together with ECC. BRUS in G-mean, COCOA and ECC in F-measure and ECC in AUC-PR are the only 4 out of 25 cases of non-significant difference between ECCRU3 and the five competing methods in the five evaluation measures. In terms of training time, BRUS has the most wins minus losses (7), followed by COCOA (4), BR (3), EBRUS and ECCRU2 (-1), ECCRU (-2), ECCRU3 (-3) 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.

To investigate the accuracy of the competing methods under different imbalance levels, we divide ImR into 7 intervals: $[1,5)$, $[5,10)$, $[10,15)$, $[15,25)$, $[25,50)$, $[50,100)$, $[100,\infty)$. Figure 3 (a) shows the percentages of the labels from all 16 datasets that fall into these intervals. We can see that more than half of the labels have $ImR \geq 100$ and only 16% of the labels have $ImR < 15$. Figure 3 (b)-(f) shows the average rank of the 8 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 $ImR \geq 15$. When $ImR < 15$ COCOA and BRUS do well in F-measure and G-mean, EBRUS does well in AUC-ROC and AUC-PR and ECC does well in F-measure, G-mean and 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 $ImR < 50$. This hints that a meta-approach selecting ECCRU for low ImR and ECCRU3 for high ImR could lead to even better results. 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.

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*	*R*	*P*	*T*	**ECC vs**	*F*	*G*	*B*	*R*	*P*	*T*
ECC	↓	↓	↓	↓	↓	↑	BR	↑	↑	↑	↑	↑	↓
BRUS	-	↓	↓	↓	↑	↓	BRUS	↑	↓	-	↑	↑	↓
EBRUS	-	↓	↓	↓	↓	-	EBRUS	↑	-	-	↓	↑	↓
COCOA	-	↓	↓	↓	↓	-	COCOA	-	-	-	↓	↑	↓
ECCRU	-	↓	↓	↓	↓	↑	ECCRU	-	↓	↓	↓	↑	↓
ECCRU2	↓	↓	↓	↓	↓	↑	ECCRU2	-	↓	↓	↓	↑	↓
ECCRU3	↓	↓	↓	↓	↓	↑	ECCRU3	-	↓	↓	↓	-	↓
BRUS vs	*F*	*G*	*B*	*R*	*P*	*T*	**EBRUS vs**	*F*	*G*	*B*	*R*	*P*	*T*
BR	-	↑	↑	↑	↓	↑	BR	-	↑	↑	↑	↑	-
ECC	↓	↑	-	↓	↓	↑	ECC	↓	-	-	↑	↓	↑
EBRUS	-	↑	-	↓	↓	↑	BRUS	-	↓	-	↑	↑	↓
COCOA	-	↑	↓	↓	↓	↑	COCOA	-	↓	-	-	-	↓
ECCRU	-	-	↓	↓	↓	↑	ECCRU	-	↓	↓	-	↓	-
ECCRU2	↓	-	↓	↓	↓	↑	ECCRU2	↓	↓	↓	-	↓	-
ECCRU3	↓	-	↓	↓	↓	↑	ECCRU3	↓	↓	↓	↓	↓	-
COCOA vs	*F*	*G*	*B*	*R*	*P*	*T*	**ECCRU vs**	*F*	*G*	*B*	*R*	*P*	*T*
BR	-	↑	↑	↑	↑	-	BR	-	↑	↑	↑	↑	↓
ECC	-	-	-	↑	↓	↑	ECC	-	↑	↑	↑	↓	↑
BRUS	-	↓	↑	↑	↑	↓	BRUS	-	-	↑	↑	↑	↓
EBRUS	-	↑	-	-	-	↑	EBRUS	-	↑	↑	-	↑	-
ECCRU	-	↓	↓	↓	-	↑	COCOA	-	↑	↑	↑	-	↓
ECCRU2	-	↓	↓	↓	-	↑	ECCRU2	-	-	-	-	-	-
ECCRU3	-	↓	↓	↓	↓	↑	ECCRU3	↓	-	-	↓	↓	-
ECCRU2 vs	*F*	*G*	*B*	*R*	*P*	*T*	**ECCRU3 vs**	*F*	*G*	*B*	*R*	*P*	*T*
BR	↑	↑	↑	↑	↑	↓	BR	↑	↑	↑	↑	↑	↓
ECC	-	↑	↑	↑	↓	↑	ECC	-	↑	↑	↑	-	↑
BRUS	↑	-	↑	↑	↑	↓	BRUS	↑	-	↑	↑	↑	↓
EBRUS	↑	↑	↑	-	↑	-	EBRUS	↑	↑	↑	↑	↑	-
COCOA	-	↑	↑	↑	-	↓	COCOA	-	↑	↑	↑	↑	↓
ECCRU	-	-	-	-	-	-	ECCRU	↑	-	-	↑	↑	-
ECCRU3	-	-	↓	↓	↓	↑	ECCRU2	-	-	↑	↑	↑	↓

5. Conclusion

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 levels of exploitation of majority examples, leading to the ECCRU2 and ECCRU3 algorithms. Our empirical study showed that the proposed method 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.

Acknowledgments

Bin Liu is supported from the China Scholarship Council (CSC) under the Grant CSC No.201708500095. Grigorios Tsoumakas is partially supported from Atypon Systems LLC.

References

Josephine S Akosa. Predictive Accuracy : A Misleading Performance Measure for Highly Imbalanced Data Classified negative. *SAS Global Forum*, 2017.

Alessio Benavoli, Giorgio Corani, and Francesca Mangili. Should We Really Use Post-Hoc Tests Based on Mean-Ranks? *Journal of Machine Learning Research*, 17:1–10, 2016. ISSN 15337928.

Matthew R. Boutell, Jiebo Luo, Xipeng Shen, and Christopher M. Brown. Learning multi-label scene classification. *Pattern Recognition*, 37(9):1757–1771, 2004. ISSN 00313203. doi: 10.1016/j.patcog.2004.03.009.

L Breiman, J H Friedman, R A Olshen, and C J Stone. *Classification and Regression Trees*, volume 19. 1984. ISBN 0412048418. doi: 10.1371/journal.pone.0015807.

Peng Cao, Xiaoli Liu, Dazhe Zhao, and Osmar 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)*, pages 244–255, Cham, 2017. Springer International Publishing. ISBN 978-3-319-52941-7.

Francisco Charte, Antonio Rivera, Mara Jos del Jesus, and Francisco 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)*, volume 8073 LNAI, pages 150–160, 2013. ISBN 9783642408458. doi: 10.1007/978-3-642-40846-5{_}16.

Francisco Charte, Antonio J. Rivera, Mara J. Del Jesus, and Francisco Herrera. MLeNN: A first approach to heuristic multilabel undersampling. In *Intelligent Data Engineering and Automated Learning – IDEAL 2014*, volume 8669 LNCS, pages 1–9. Springer International Publishing, 2014. ISBN 9783319108391. doi: 10.1007/978-3-319-10840-7{_}1.

Francisco Charte, Antonio Rivera, Mara Jos Del Jesus, and Francisco Herrera. Resampling multilabel datasets by decoupling highly imbalanced labels. In *Lecture Notes in Artificial Intelligence (Subseries of Lecture Notes in Computer Science)*, volume 9121, pages 489–501, 2015a. ISBN 9783319196435. doi: 10.1007/978-3-319-19644-2{_}41.

Francisco Charte, Antonio J. Rivera, Mara J. del Jesus, and Francisco Herrera. Addressing imbalance in multilabel classification: Measures and random resampling algorithms. *Neurocomputing*, 163:3–16, 9 2015b. ISSN 09252312. doi: 10.1016/j.neucom.2014.08.091.

Francisco Charte, Antonio J. Rivera, Mara J. Del Jesus, and Francisco Herrera. MLSMOTE: Approaching imbalanced multilabel learning through synthetic instance generation. *Knowledge-Based Systems*, 89:385–397, 2015c. ISSN 09507051. doi: 10.1016/j.knosys.2015.07.019.

Ken Chen, Bao-Liang Lu, and James 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*, pages 1770–1775. IEEE, 2006. ISBN 0-7803-9490-9. doi: 10.1109/IJCNN.2006.246893.

Zachary Alan Daniels and Dimitris N. Metaxas. Addressing Imbalance in Multi-Label Classification Using Structured Hellinger Forests. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence*, pages 1826–1832, 2017.

Krzysztof Dembczyński, Weiwei Cheng, and Eyke Hüllermeier. Bayes Optimal Multilabel Classification via Probabilistic Classifier Chains. *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 279–286, 2010.

Sareewan Dendamrongvit and Miroslav Kubat. Undersampling Approach for Imbalanced Training Sets and Induction from Multi-label Text-Categorization Domains. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, pages 40–52. 2010. ISBN 3642146392. doi: 10.1007/978-3-642-14640-4{_}4.

Salvador Garcia and Francisco Herrera. An Extension on "Statistical Comparisons of Classifiers over Multiple Data Sets" for all Pairwise Comparisons. *Journal of machine learning research*, 9:2677–2694, 2008. ISSN 1532-4435.

Haibo He and Edwardo A. Garcia. Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering*, 21(9):1263–1284, 2009. ISSN 10414347. doi: 10.1109/TKDE.2008.239.

Cunhe Li and Guoqiang Shi. Improvement of learning algorithm for the multi-instance multi-label RBF neural networks trained with imbalanced samples. *Journal of Information Science and Engineering*, 2013. ISSN 10162364.

Li Li and Houfeng Wang. Towards Label Imbalance in Multi-label Classification with Many Labels. *CoRR*, abs/1604.0, 2016.

Jesse Read, Bernhard Pfahringer, Geoff Holmes, and Eibe Frank. Classifier chains for multi-label classification. *Machine Learning*, 85(3):333–359, 2011. ISSN 08856125. doi: 10.1007/s10994-011-5256-5.

Timothy N. Rubin, America Chambers, Padhraic Smyth, and Mark Steyvers. Statistical topic models for multi-label document classification. *Machine Learning*, 88(1-2):157–208, 2012. ISSN 08856125. doi: 10.1007/s10994-011-5272-5.

Konstantinos Sechidis, Grigorios Tsoumakas, and Ioannis Vlahavas. On the Stratification of Multi-label Data. In *Proc. 2011 European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 145–158. Springer Berlin Heidelberg, Athens, Greece, 2011.

Konstantin Sozykin, Adil Mehmood Khan, Stanislav Protasov, and Rasheed Hussain. Multi-label Class-imbalanced Action Recognition in Hockey Videos via 3D Convolutional Neural Networks. *CoRR*, abs/1709.0, 2017.

Eleftherios Spyromitros-Xioufis, Grigorios Tsoumakas, William Groves, and Ioannis Vlahavas. Multi-target regression via input space expansion: treating targets as inputs. *Machine Learning*, 104(1):55–98, 2016. ISSN 15730565. doi: 10.1007/s10994-016-5546-z.

Kai Wei Sun and Chong Ho Lee. Addressing class-imbalance in multi-label learning via two-stage multi-label hypernetwork. *Neurocomputing*, 2017. ISSN 18728286. doi: 10.1016/j.neucom.2017.05.049.

Muhammad Atif Tahir, Josef Kittler, and Fei Yan. Inverse random under sampling for class imbalance problem and its application to multi-label classification. *Pattern Recognition*, 2012. ISSN 00313203. doi: 10.1016/j.patcog.2012.03.014.

Lei Tang, Suju Rajan, and Vijay K. Narayanan. Large scale multi-label classification via metalabeler. In *Proceedings of the 18th international conference on World wide web - WWW '09*, page 211, 2009. ISBN 9781605584874. doi: 10.1145/1526709.1526738.

Gorn Tepvorachai and Chris Papachristou. Multi-label imbalanced data enrichment process in neural net classifier training. In *Proceedings of the International Joint Conference on Neural Networks*, 2008. ISBN 9781424418213. doi: 10.1109/IJCNN.2008.4633966.

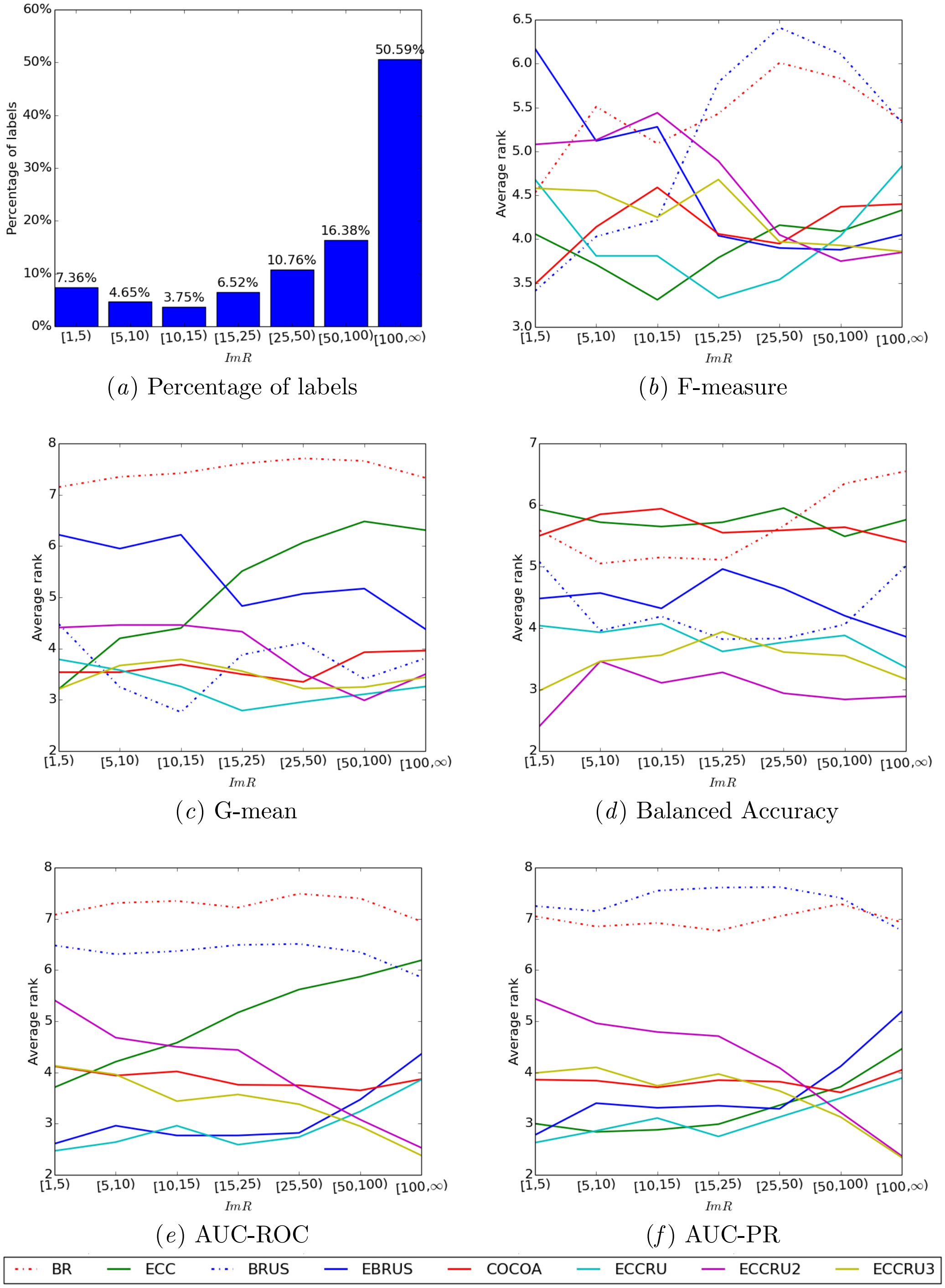
G. Tsoumakas, E. Spyromitros-Xioufis, J. Vilcek, and I. Vlahavas. MULAN: A Java library for multi-label learning. *Journal of Machine Learning Research*, 2011. ISSN 1532-4435.

Shixiang Wan, Yucong Duan, and Quan Zou. HPSLPred: An Ensemble Multi-Label Classifier for Human Protein Subcellular Location Prediction with Imbalanced Source. *Proteomics*, 2017. ISSN 16159861. doi: 10.1002/pmic.201700262.

Baoyuan Wu, Siwei Lyu, and Bernard Ghanem. Constrained Submodular Minimization for Missing Labels and Class Imbalance in Multi-label Learning. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI'16, pages 2229–2236. AAAI Press, 2016.

Wenrong Zeng, X Chen, and Hong Cheng. Pseudo labels for imbalanced multi-label learning. In *2014 International Conference on Data Science and Advanced Analytics (DSAA)*, pages 25–31, 10 2014. doi: 10.1109/DSAA.2014.7058047.

Min-Ling Zhang, Yu-Kun Li, and Xu-Ying Liu. Towards class-imbalance aware multi-label learning. In *Proceedings of the 24th International Conference on Artificial Intelligence*, pages 4041–4047, 2015. ISBN 9781577357384.



(a) Percentage of labels

(b) F-measure

(c) G-mean

(d) Balanced Accuracy

(e) AUC-ROC

(f) AUC-PR

Figure 3: 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.