On the Thresholding Strategy for Infrequent Labels in Multi-label Classification

Yu-Jen Lin
National Taiwan University
r10922a14@ntu.edu.tw

Chih-Jen Lin
National Taiwan University
cjlin@csie.ntu.edu.tw
Mohamed bin Zayed University of Artificial Intelligence
chihjen.lin@mbzuai.ac.ae

ABSTRACT

In multi-label classification, the imbalance between labels is often a concern. For a label that seldom occurs, the default threshold used to generate binarized predictions of that label is usually sub-optimal. However, directly tuning the threshold to optimize F-measure has been observed to overfit easily. In this work, we explain why this overfitting occurs. Then, we analyze the FBR heuristic, a previous technique proposed to address the overfitting issue. We explain its success but also point out some problems unobserved before. Then, we first propose a variant of the FBR heuristic that not only fixes the problems but is also more justifiable. Second, we propose a new technique based on smoothing the F-measure when tuning the threshold. We theoretically prove that, with proper parameters, smoothing results in desirable properties of the tuned threshold. Based on the idea of smoothing, we then propose jointly optimizing micro-F and macro-F as a lightweight alternative free from extra hyperparameters. Our methods are empirically evaluated on text and node classification datasets. The results show that our methods consistently outperform the FBR heuristic.

CCS CONCEPTS

• **Computing methodologies → Machine learning**.

KEYWORDS

multi-label classification, F-measure, infrequent labels, threshold adjustion

ACM Reference Format:
Yu-Jen Lin and Chih-Jen Lin. 2023. On the Thresholding Strategy for Infrequent Labels in Multi-label Classification. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management (CIKM '23), October 21–25, 2023, Birmingham, United Kingdom.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3583780.3614996

1 INTRODUCTION

In multi-label classification, the goal is to predict a set of relevant labels given an input instance. Applications of this setting, to name a few, include text classification [23], product search or recommendation [2], medical code prediction [17], and node classification [5, 19, 26].

Among the many applications of multi-label classification, the number of labels involved ranges from tens to millions or more. Depending on the number of labels and the specific application, different ways of prediction are applied, which are then evaluated with different metrics. Some applications like recommender systems deal with hundreds of thousands of labels while focusing on predicting a few top labels correctly. In such situations, metrics like Precision@k or nDCG@k are common choices [30, 32]. However, there are still many important applications with a smaller number of labels (e.g., $\leq 10,000$), such as node classification [5, 19, 26] or medical code prediction [17]. For these tasks, we aim to predict all the relevant labels for an instance instead of only the top ones. In these cases, metrics such as micro-F and macro-F are the most common. In this work, we consider applications that require predicting all relevant labels and investigate the techniques for optimizing micro-F and macro-F.

An important issue in multi-label classification is the imbalance between labels. As the number of labels exceeds hundreds or more, there may be labels that occur in only a few samples. Learning a robust predictor for these labels thus poses a significant challenge. For the convenience of our discussion, we roughly separate the labels into rare (less than 10 positive samples), medium (between 10 and 100 samples) and frequent labels (more than 100 samples). We use infrequent labels to refer to rare and medium labels, which are our main focus in this work. Among the two metrics mentioned above, macro-F is known to take infrequent labels into account more effectively [15, 22]. Therefore, we use macro-F as the main metric to judge a classifier's performance on infrequent labels.

In addition to the choice of metrics, standard machine learning algorithms must be adapted to imbalanced datasets. Common techniques include over/under-sampling [6, 8], cost-sensitive methods [6, 8], and threshold adjustion [10, 11, 25]. Currently, one of the most effective ways to optimize macro-F is by tuning a separate threshold for each label. The tuned thresholds are then applied to the decision values of labels to give predictions. A pioneer in this direction is SCut [29]. However, applying SCut to infrequent labels has been observed to overfit easily, thus posing a challenging research problem [9, 29]. As far as we know, not much work has provided a good understanding of why the SCut method overfits so badly, and so far, the FBR heuristic [29] is the only technique to battle this issue. Despite FBR's success in the past [4, 12], it is a heuristic that is not well-understood. Therefore, in this work, we aim to analyze the difficulties of tuning thresholds for infrequent labels and provide principled solutions to the problems discovered.

In previous works on thresholding, the classifier for each label is usually a linear classifier [4, 12]. We follow them to consider linear support vector machines (SVM). However, our method should also apply to other models (e.g., neural networks) as long as the prediction is generated by thresholding on the scores of each label.

Our contribution can be summarized as follows:

- We explain that when optimizing F-measures (and similar metrics) of infrequent labels by tuning the threshold, overfitting occur due to how these metrics are formulated.
- The FBR heuristic [29], a previous technique proposed to solve the overfitting problem, is thoroughly analyzed. We are likely the first to explain why this heuristic is effective. Further, we novelly point out some problems of the heuristic.
- To solve the problems discovered, we first propose a variant of the FBR method. Next, we propose using a smoothed F-measure as a surrogate when tuning thresholds. In contrast to the FBR heuristic, our methods are more principled and have theoretical justifications.
- Both FBR and our proposed methods require a two-level cross-validation procedure, which may be time-consuming for large problems. We propose a lightweight version of our method by jointly optimizing micro-F and macro-F. The resulting method is much more efficient, requiring only one level of cross-validation.
- Evaluations on text and node classification show that our methods effectively improve upon the FBR heuristic.

The remaining work is organized as follows: In Section 2, we formally define the problem and review some previous works. In Section 3, we dive deeper into the FBR heuristic to explain why it was successful. We further point out that FBR makes pessimistic decisions in some situations and propose a solution to the problem. In Section 4, we introduce smoothing for regularizing threshold tuning and propose several algorithms based on this idea. Finally, we present experimental results in Section 5 and conclude this study in Section 6.

Supplementary materials and the experiment program are available at https://www.csie.ntu.edu.tw/~cjlin/papers/thresholding/. The method jointly optimizing micro-F and macro-F, proposed in Section 4.3, is now the thresholding strategy provided in our multi-label classification package LibMultiLabel[1]. This work is an extension of the first author's master thesis [14].

2 PRELIMINARIES

2.1 Problem Setting

In multi-label classification, we are given a set of target labels $Y = \{1, 2, \dots, L\}$ and a set of training samples $\{(x_i, y_i)\}$, where x_i is from a feature domain X and y_i is a vector in $\{-1, +1\}^L$. A value of +1 in the jth position of y_i indicates that label j is relevant to x_i, while -1 indicates an irrelevant label. Our goal is to train a model $\Phi(x) : X \longrightarrow \{-1, +1\}^L$ using the training samples, such that it correctly predicts the set of relevant labels given any sample x.

The model Φ can be decomposed into L components:

$$\Phi_j : X \longrightarrow \{-1, +1\} \text{ for } j = 1, 2, \dots, L.$$

[1]https://www.csie.ntu.edu.tw/~cjlin/libmultilabel/

Each component Φ_j is then trained to predict whether label j is relevant to a sample, which is a binary classification problem. The components Φ_j are often implemented by a scoring function $\phi_j : X \longrightarrow \mathbb{R}$ outputting a decision value for each label j. Then, a threshold T is applied to $\phi_j(x)$ to generate binarized predictions:

$$\Phi_j(x) = \begin{cases} +1 & \phi_j(x) > T, \\ -1 & \phi_j(x) \le T. \end{cases} \tag{1}$$

When Φ_j and ϕ_j are solved independently for each j as a binary classification problem, this approach is called binary relevance. In contrast, methods like neural networks typically train the components for all j together. This work mainly focuses on the binary relevance method applied to linear support vector machines (SVM). That is, the scoring function is in the form

$$\phi_j(x) = w_j^T x + b_j,$$

where w_j and b_j are learned parameters of the SVM. The default threshold for SVMs is usually set at $T = 0$. Besides SVM, other models can be considered as long as a score is assigned to each label and thresholded as in (1). Examples include applying binary relevance to other base classifiers or neural networks that output a probability for each label, which is usually thresholded at $T = 0.5$.

2.2 Evaluation Metrics

As mentioned in the introduction, we focus our discussion on micro-F and macro-F, which are common extensions of F-measure for multi-label problems. In the setting of binary classification, the F-measure of a binary prediction is defined as

$$\text{F} = \frac{2\text{tp}}{2\text{tp} + \text{fp} + \text{fn}}, \tag{2}$$

where tp, fp and fn stand for the number of true positives, false positives, and false negatives, respectively.

When multiple labels exist, F-measure is extended in mainly two ways. Macro-F is the average F-measure over all labels, while micro-F calculates a single F-measure using counts of tp, fp and fn summed over all labels. Let tp_j, fp_j and fn_j denote the respective counts calculated only using label j. Macro-F and micro-F can be expressed as:

$$\text{F}^{\text{micro}} = \frac{\sum_{j=1}^{L} 2\text{tp}_j}{\sum_{j=1}^{L} (2\text{tp}_j + \text{fp}_j + \text{fn}_j)},$$

$$\text{F}^{\text{macro}} = \frac{1}{L} \sum_{j=1}^{L} \text{F}_j, \text{ where } \text{F}_j = \frac{2\text{tp}_j}{2\text{tp}_j + \text{fp}_j + \text{fn}_j}.$$

In the next section, we review some approaches proposed to optimize F-measure.

2.3 Previous Works

Standard machine learning algorithms usually assume the positive and negative classes to be balanced, and they optimize measures like accuracy that treat each class equally. In this case, using default thresholds described in Section 2.1 works reasonably well. However, for an infrequent label, the corresponding binary problem is imbalanced, with only a few positives. In such a situation, not tuning the threshold can be a "critical mistake" [21]. Several works have

Algorithm 1 SCut (one label)

```
Input: A set of samples $D = \{(x_i, y_i) \mid y_i \in \{-1, +1\}\}$, a metric $f(T)$, number of folds $V$
Output: A trained scoring function $\phi$, a threshold $T$
1: Randomly partition $D$ to $V$ subsets $D_1, D_2, \ldots, D_V$
2: for $k = 1, 2, \ldots, V$ do
3:     Train a model $\phi^{(k)}$ on $D \setminus D_k$
4:     On the validation set $D_k$, calculate the decision values $Z = \{\phi^{(k)}(x) \mid x \in D_k\}$.
5:     Find the threshold $T^{(k)}$ that maximizes $f(T^{(k)})$ on $Z$.
6: end for
7: $T = \frac{1}{V} \sum_{k=1}^{V} T^{(k)}$
8: Train a $\phi$ on $D$
9: Return $(\phi, T)$
```

demonstrated the importance of threshold tuning for traditional machine learning models [1, 4, 25, 29] and neural networks [10].

2.3.1 SCut. Instead of using the default threshold T in (1), SCut [29] has a separate threshold T_j for each label. The thresholds are usually tuned to maximize a chosen metric on a validation set that was not used to train the scoring function. A single validation set or a V-fold cross-validation (CV) can be used. When a CV is used, the average of the V thresholds is used as the final threshold. After that, a final model is re-trained with all training data, for the extra data (especially positive samples) from the validation set often improves the model further. Since the methods proposed in this work are all based on SCut, we give the procedure for tuning the threshold of one label in Algorithm 1. We define the algorithm in a general form such that any evaluation function $f(T)$ can be used to select the threshold. We provide the details about finding the optimal threshold in step 5 of Algorithm 1. Let N be the size of the validation set Z. For simplicity, we assume all the decision values $\phi^{(k)}(x_i) \in Z$ are all distinct. First, we sort the decision values in Z, so that we have $\phi^{(k)}(x_i) < \phi^{(k)}(x_{i+1})$ for $i = 1, \ldots, N-1$. Then the optimization of the measure f is done by searching through these candidate thresholds:

$$(\tau_0, \tau_1, \ldots, \tau_N) \text{ where}$$
$$\tau_i = \begin{cases} \phi^{(k)}(x_1) - \epsilon & i = 0, \\ \frac{1}{2}\left(\phi^{(k)}(x_i) + \phi^{(k)}(x_{i+1})\right) & 0 < i < N, \\ \phi^{(k)}(x_N) + \epsilon & i = N, \end{cases} \tag{3}$$
$$\text{for some small constant } \epsilon > 0.$$

When our goal is to optimize macro-F, it suffices to apply Algorithm 1 with F-measure to each label since macro-F is the average of F-measures for each label. This process is the SCut method from [29]. In contrast, different labels are not separable when optimizing micro-F. Therefore, a cyclic optimization over the labels is needed to achieve an optimal solution [4, 20] (but a single pass over the labels may be enough to produce a good model). When tuning the threshold of label j, the tp, fp and fn for other labels are treated as constants, and the resulting micro-F as a function of T_j is used to select the threshold.

SCut given in Algorithm 1 assumes that the score functions ϕ_j can be independently trained as in binary relevance. To apply the idea to neural networks where the output unit for each label is trained together, we can modify Algorithm 1 so that all output units are trained together on the training set in step 3. However, the threshold of each label is separately tuned on the validation sets and averaged in step 5 and step 7, respectively.

2.3.2 Cost-based. Cost-sensitive learning is a general technique for handling imbalanced data based on modifying the costs of false positives and false negatives. Moreover, a previous study theoretically showed that cost-sensitive methods can also optimize F-measure [18]. In the case of SVMs, for each binary subproblem, the weight C^+ for the loss of positive samples is selected via CV [1, 12, 13]. However, several studies showed that, for the particular case of applying SVM to text classification, adjusting the threshold is more effective than tuning the costs [1, 12, 25].

2.3.3 FBR Heuristic. While SCut is a reasonable approach, it has been discovered to overfit easily on rare labels [9, 29]. Surprisingly, with rare labels being a common problem in multi-label classifications, we have found little work dedicated to deal with this overfitting problem. An exception is the FBR heuristic [29]. In this approach, a value $fbr \in (0, 1)$ is pre-specified. If at step 5 of Algorithm 1, the maximal F-measure on the validation set does not exceed fbr, the FBR heuristic sets the threshold $T^{(k)}$ to $\max(Z)$, the highest decision value in the validation set. This method, referred to as SCutFBR.1 in [29], was shown effective [4, 12]. In practice, the parameter fbr is selected with a CV procedure wrapping Algorithm 1. Because Algorithm 1 already has its inner-level CV for a selecting threshold, there are two levels of CV in total.

In addition to SCutFBR.1, the same author of [29] proposed a less-used variant of the method, called SCutFBR.0, that sets the threshold to infinity instead of $\max(Z)$. Judging from this clue, the authors likely intended to increase the threshold so that the number of false positives in the testing phase is decreased.

Although FBR is currently the best method for mitigating the overfitting of SCut, it is a heuristic without good explanations. In the next section, we dive deeper into the FBR heuristic to explain its success but also point out its problems.

3 WHY FBR WORKS AND ITS ISSUES

To understand how FBR benefits the prediction of rare labels, we first examine an interesting behavior of the heuristic in Section 3.1. It shows that FBR actually improves the predictions of rare labels by lowering the threshold instead of increasing it as the authors likely intended. Then, in Section 3.2, we explain why lowering the threshold is the correct strategy. Finally, we point out some issues of FBR in Section 3.3, which we believe were not found before. We provide the solutions to these problems in Section 3.4 and Section 4.

3.1 An Interesting Behavior of FBR

A previous study [4] pointed out that for rare labels, the FBR heuristic shows an interesting behavior of choosing a threshold lower than the default $T = 0$. This behavior, associated with improving macro-F, occurs when the validation set consists of only negative

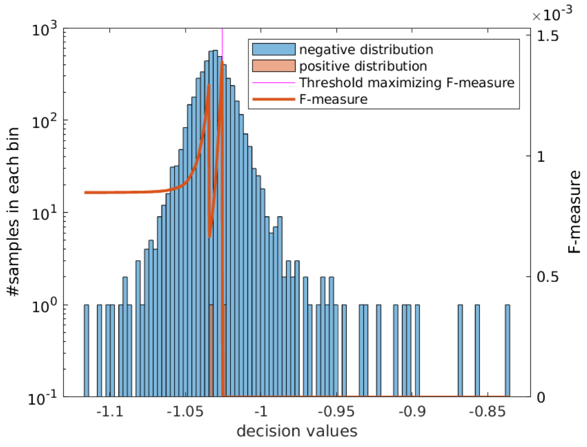

Figure 1: The distribution of the decision values from one of the validation sets of a binary subproblem from Wiki10-31K. The plotted F-measure is calculated by setting the threshold to the mid-point between adjacent decision values and counting the corresponding tp, fp and fn. In this case, the optimal value of F-measure clearly occurs at a too-low threshold, which results in many false positives. Also, notice that the gain in F-measure is almost zero (1.4×10^{-3}).

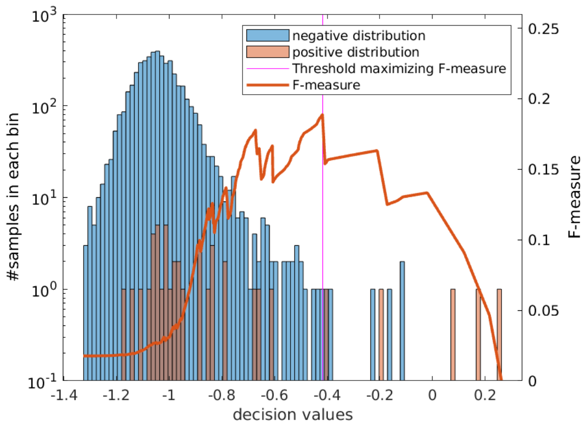

Figure 2: The distribution of the decision values from one of the validation sets of a binary subproblem from Wiki10-31K.

samples, a common situation for rare labels. Since there are no positive samples and thus no true positive predictions, the F-measure on the validation set is always 0, which is smaller than fbr. As a result, the FBR heuristic sets the threshold to the highest decision value of the validation set. Since the validation set contains only negative samples, the highest decision value is often lower than the original threshold at 0.

In addition to the situation pointed out by [4], we further discovered that FBR may still give a threshold lower than 0 when there are positive validation samples. When the number of positive training samples is too few to represent the full positive distribution, the positive validation samples are often predicted as negative [1]. We give an illustration in Figure 1. In this case, optimizing the F-measure on the validation decision values leads to a too-low threshold that gives many false positives. This behavior can be understood from the definition of F-measure in Equation (2). When tp = 0, F-measure is always zero, regardless of the value of fp. Therefore, any threshold that gives tp $\neq$ 0, no matter how large fp is, would give a non-zero F-measure and be deemed better than a reasonable threshold with tp = 0. In Supplementary E, we perform experiments to show that situations like Figure 1 frequently occur in practice.

The example above explains why SCut, which optimizes the F-measure on each label, easily overfits on rare labels. In this example, the FBR heuristic would detect the low F-measure attained and set the threshold to the negative distribution's highest value. The resulting threshold would avoid many false positives and still tends to be lower than the original threshold 0.

We note that the same overfitting issue occurs in other metrics that stay at 0 when tp = 0. They include recall, precision, F_β and G-mean. The threshold optimization for these metrics should be proceeded with care when the positive samples are rare.

3.2 The Benifit of a Threshold Lower than the Default Value

We have seen that, for rare labels, FBR has an interesting behavior of lowering the threshold to below zero. This property is likely unknown to the original author of [29]. In this section, we explain why this behavior is desirable when the positive samples are rare.

In imbalanced scenarios, it has been known that the decision boundaries learned by common algorithms are often skewed toward the minority, which results in under-predicting the minority class. This behavior occurs because the positive samples are too few to fully specify the positive distribution boundary. A synthetic example for SVMs demonstrating this behavior was given in [28]. Furthermore, [1, 25] tested SVMs on real-world text data and discovered that SVM's default threshold is too high, misclassifying many positive samples as negative. In recent years, similar problems have also been discovered on neural networks, showing that the default probability threshold 0.5 is far from optimal [10].

The question, then, is how much to lower the threshold. From the discussion in Section 3.1, we know the decision should not rely on optimizing F-measure. In [1], they propose to estimate a probabilistic model $p(y = +1 \mid w^T x + b)$ from the training set and threshold the probability at some small value (e.g., 0.02). This approach usually lowers the threshold and is shown to be effective for infrequent labels. However, it results in poor performance for frequent labels [1].

Our idea is similar to that of [1]. We argue that when the positive samples are infrequent, the threshold should be placed on the upper bound of the negative distribution estimated on a validation set. In other words, the algorithm turns into an outlier-detection-like classifier. Since the upper bound of the negative distribution is estimated on the validation set, this boundary should not give too many false positives in the testing phase while being able to detect some of the positive samples. These successfully-detected positive samples then improve the F-measure. As for how to lower the threshold automatically for infrequent labels without affecting the performance of frequent labels, we provide solutions in Section 3.4 and Section 4.

Note that FBR sets the threshold to the highest decision value in the validation set instead of the highest decision value in the negative samples. These two settings coincide when there are no positive samples, or all positive samples are mixed with the negatives (like Figure 1), which are common when performing CV on rare labels. This coincidence is probably why FBR successfully improves the F-measure on rare labels. However, in the next section we argue that FBR is an inferior strategy since it can be too pessimistic sometimes.

3.3 Issues of FBR Heuristic

We have explained why FBR works for rare labels. However, we point out some issues of FBR in other situations. In Section 3.2, we mentioned that FBR sets the threshold to the highest decision value of all validation samples, instead of only the negative samples. We now show that FBR can be too pessimistic for medium labels (which are slightly more frequent than rare labels, as defined in Section 1). We give an example validation set of a medium label in Figure 2. In this case, there are three positive samples on the right. We call these positive samples above all negative samples the **"easy positives"**, and we argue that a reasonable threshold should not give up the easy positives. However, because the F-measure is never higher than 0.2 for any threshold, if $fbr = 0.2$, the FBR algorithm would set the threshold higher than the easy positives, losing performance that can be easily grasped. Instead, setting the threshold slightly higher than the highest negative sample seems better because it correctly predicts the easy positives and avoids many false positives.

Situations like Figure 2 tend to happen for medium labels because there are enough positive samples for easy positives to appear but not enough to fully specify the positive distribution, which leads to a low validation F-measure. In Supplementary E, we experimentally show that situations like Figure 2 indeed occur in practice.

3.4 A New and More Explainable Variant of SCutFBR

Previous discussions in Section 3 show that when situations like Figure 1 occur (i.e., the label is rare and the positive validation samples are mixed in the negative distribution), the attained F-measure is usually low. Therefore, we can check if the attained F-measure is lower than a cutting point to detect a bad threshold. Since such a threshold cannot be trusted, the threshold has to be reset. We then explained that it is reasonable to adjust the threshold to the highest decision values of negative samples.

To implement the preceding strategy, it is essential but difficult to judge what value of F-measure counts as too low and thus requires resetting. Therefore, one can try multiple cutting points and find the most suitable one by an outer-level cross-validation. In this regard, fbr serves as the guess of this cutting point. Based on these ideas, we can derive a more explainable variant of SCutFBR:

SCutFBR.n – The method is the same as SCut with FBR heuristic, except that the threshold is set to the **highest decision value of the negative samples** when the attained F-measure is less than fbr. A two-level cross-validation is used, where the outer level selects the parameter fbr and the inner level selects the threshold.

4 SMOOTHED F-MEASURE

In Section 3.1, we explained that, for rare labels, optimizing F-measure can lead to a too-low threshold since F-measure does not penalize false positives when $\mathrm{tp} = 0$. Besides the remedy offered in Section 3.4, we propose smoothing the F-measure by introducing some constants a and b where $a > 0$ and $b \geq 0$:

$$\mathrm{F}(\mathrm{tp}, \mathrm{fp}, \mathrm{fn}; a, b) = \frac{a + 2\mathrm{tp}}{b + 2\mathrm{tp} + \mathrm{fp} + \mathrm{fn}}. \tag{4}$$

The parameter a forces the numerator to be larger than zero, so any increase in fp (while other counts stay the same) is penalized by a decrease in the smoothed F-measure. The parameter b is included for generality and can be used to prevent dividing by zero when $\mathrm{tp} = \mathrm{fp} = \mathrm{fn} = 0$. Given a set of decision values, the counts tp, fp and fn are functions of the threshold T. Therefore, we will also use the notation $\mathrm{tp}(T)$ for the number of true positives (similarly for fp and fn) and $F(T; a, b)$ for the smoothed F-measure.

To gain more insight into smoothed F-measure, we can rewrite it as

$$\mathrm{F}(\mathrm{tp}, \mathrm{fp}, \mathrm{fn}; a, b) = \frac{a}{b + \mathrm{p} + \mathrm{tp} + \mathrm{fp}} + \frac{2\mathrm{tp}}{b + 2\mathrm{tp} + \mathrm{fp} + \mathrm{fn}} \tag{5}$$

by separating the numerator in (4) into two terms and then replace $\mathrm{tp} + \mathrm{fn}$ with p (i.e., number of positive validation samples) in the first term. We can see that the second term in (5) resembles the original F-measure (when $b = 0$, it becomes F-measure). As for the first term, because $\mathrm{tp} + \mathrm{fp}$ is the number of samples predicted as positive (i.e., the number of samples above the threshold), the denominator

$$b + \mathrm{p} + \mathrm{tp} + \mathrm{fp}$$

is nondecreasing as the threshold moves down. Therefore, a higher threshold leads to a larger value of the first term while a larger p make this term smaller. From these observations, we can interpret smoothing as adding to F-measure a term regularizing the threshold towards higher values, where the regularization automatically weakens as labels becomes more frequent (i.e., p increases).

The discussion above gives us some intuition of what smoothing does. In the next subsection, we explain more formally how smoothing solves the problems mentioned in Section 3.

4.1 How Smoothing Solves the Problems

In this section, we give in-depth discussions on the properties of smoothing to show how smoothing solves the problems mentioned in Section 3.1 and Section 3.3, respectively.

4.1.1 The Problem of Too-Low Thresholds. In Section 3.1, we mentioned that when a label is rare, the threshold maximizing F-measure is often too low, causing numerous false positives. To demonstrate how smoothing avoids such thresholds, we prove a property of smoothed F-measure in Theorem 1. For simplicity, we conduct our theoretical discussions under the following setting:

Assumption 1. Let a set of decision values along with their binary labels $D = \{(\phi(x_i), y_i) \mid y_i \in \{-1, +1\}$ for $i = 1, 2, \ldots, N\}$ be given. We assume the decision values are distinct and sorted, i.e., $\phi(x_i) < \phi(x_{i+1})$ for all $i = 1, 2, \ldots, N - 1$. Furthermore, we assume that a and b satisfy $a > 0$ and $b + \mathrm{p} > 0$, where p is the number of positive samples in D.

Theorem 1. Under Assumption 1, the threshold T^* maximizing the smoothed F-measure $F(T; a, b)$ satisfies

$$\text{fp}^* \leq \text{tp}^*(\frac{2(b+\text{p})}{a} - 1), \tag{6}$$

where fp^* and tp^* are the corresponding counts generated by thresholding at T^*.

The proof is in Supplementary A.1. Theorem 1 shows that when optimizing the smoothed F-measure, the resulting threshold never gives a too large fp unless a proportional tp is also obtained. In contrast, no such bound holds for the original F-measure. We can consider a dataset with only one positive sample. Because F-measure is always 0 when $\text{tp} = 0$, optimizing F-measure will always lower the threshold to include that one positive sample, no matter how many false positives it gives.

Furthermore, we can use Theorem 1 to show that smoothing can reset a bad threshold above the highest decision value of the negative samples when dealing with rare labels, for which these two situations commonly occur:

(1) There are no positive samples in the validation set.
(2) There are positive samples in the validation set. However, due to the lack of positive training samples, the positive validation samples are predicted as negative as in Figure 1.

In the first situation, since $\text{tp} = 0$ for every threshold, we must have $\text{tp}^* = 0$. Theorem 1 then tells us fp^* must also be zero. Therefore, the resulting threshold is above the highest negative sample. In the second situation, a threshold lower than the highest decision value of negative samples can easily generate a large fp with a small tp. Under suitable (a, b), such a threshold would not satisfy (6) and thus not be selected when optimizing smoothed F-measure. For example, in Figure 1, we have $\text{p} = 2$, and the threshold maximizing the original F-measure gives $\text{tp} = 1, \text{fp} = 1419$. If we choose $a = b = 1$ and plug all number into (6), we see that

$$1419 \leq 1 \cdot (\frac{2(1+2)}{1} - 1)$$

does not hold. Therefore, the threshold maximizing the smoothed F-measure would be above the highest negative sample in this case. The discussion above shows how smoothing solves the problem of too-low thresholds from Section 3.1.

4.1.2 *The Problem of Pessimistic Thresholds.*

In Section 3.3, we pointed out FBR selects thresholds that are too pessimistic, giving up the easy positives. We now show that smoothed F-measure does not give up the easy positives.

Theorem 2. Under Assumption 1, if $a < 2(b + \text{p})$ and there exists i' such that

$$y_i = +1 \text{ for all } i \geq i', \tag{7}$$

then the threshold T^* maximizing $F(T; a, b)$ satisfies $T^* < \phi(x_{i'})$.

The proof is in Supplementary A.2. Theorem 2 shows that if index i' is an easy positive sample, then the threshold optimizing the smoothed F-measure will be lower than sample i' as long as $a < 2(b + \text{p})$.

We have finished explaining how smoothing solves the problems discovered in Section 3. Next, we discuss in detail how a and b may be selected in practice.

4.2 Selecting a and b

As shown in Theorem 1, a and b control the allowed trade-off between fp and tp. Intuitively, (a, b) that are too large may cause the smoothed F-measure to deviate from the original F-measure or affect the performance on frequent labels. Picking the right parameter is not easy. Therefore, like what we did in Section 3.4 for choosing the cutting point, we apply CV to select a suitable (a, b). Specifically, we consider a list of candidate values to search over:

$$(a_1, b_1), (a_2, b_2), \ldots, (a_m, b_m).$$

Notice that we do not use a list of values for a and b separately and search over all combinations. The reason is that the proper range of a depends on b, as we will now derive. For the upper bound of a, we use the following corollary of Theorem 1.

Corollary 3. Under the same assumptions in Theorem 1, if $a > 2(b+\text{p})$, then the threshold T^* maximizing the smoothed F-measure is always higher than all decision values.

The proof is in Supplementary A.3. Clearly, unconditionally placing the threshold higher than all decision values would give up the easy positives, which we argued to be too pessimistic in Section 3.3. Therefore, $2(b+\text{p})$ can be an upper bound for a since any value larger gives an undesirable behavior. Moreover, $a < 2(b + \text{p})$ allows Theorem 2 to hold, which guarantees that the resulting threshold does not give up any easy positive.

Next, we derive an lower bound for a with the following theorem:

Theorem 4. Under Assumption 1, if a and b also satisfy

$$a < \frac{2(b+\text{p})}{N},$$

then the threshold T^* maximizing the smoothed F-measure is always less than or equal to the largest decision value among the positive samples.

The proof is in Supplementary A.4. Recall that the original purpose of smoothing is to prevent lowering the threshold to always include at least one positive. Theorem 4 shows that if a is too small, the behavior of including at least one positive sample would always occur. Therefore, $2(b + \text{p})/N$ serves as a lower bound for a.

We do not derive a bound for b. Intuitively, it should not be too large. Otherwise, the smoothed F-measure would be too distorted to act as a surrogate for the F-measure. Also, the parameter b should not grow with the number of positive samples since smoothing is for stabilizing the threshold optimization of infrequent labels, and the threshold optimization of frequent labels is already stable without smoothing. In practice, we can prepare a prespecified geometric sequence for b. Then, for each b, we select a as a geometric sequence within the range derived in Corollary 3 and Theorem 4. Empirically, we found that searching b up to 100 is adequate.

Knowing the proper search range for the parameters, we devise the following methods for selecting a and b and optimizing macro-F:

smooth-each – For each label j, this method performs a two-level CV. Under a fixed (a, b), the inner level selects the threshold using Algorithm 1 with $F(T; a, b)$ as the metric. The outer level checks a list of (a, b) pairs and selects the one achieving the best validation F-measure. Using the best (a^*, b^*), a final model for label j is produced by running Algorithm 1 with all training data. In short, a and b are selected independently for each label.

smooth-all – This method performs a two-level CV. For all labels together, the outer-level CV selects a pair of (a, b) that achieves the best validation macro-F. The inner level uses the same given (a, b) to select a threshold for each label using Algorithm 1 with $F(T; a, b)$. Then, we use the best (a^*, b^*) and the entire training set to run Algorithm 1 again for each label.

Note that in the outer-level CV, we use F-measure and macro-F, instead of smoothed versions, for selecting (a, b) in smooth-each and smooth-all, respectively. The reason is that the original macro-F is the ultimate goal we care about, and the metric used in selecting (a, b) should be independent of the (a, b) being selected.

The above procedures both involve a CV process for selecting (a, b). Under each training/validation split in the outer-level CV, and some given (a, b), a call to Algorithm 1 performs an inner-level CV, which trains $V+1$ models in steps 3 and 8. Suppose we search m pairs of (a, b) by calling Algorithm 1 for each pair, then $m(V + 1)$ models are trained, which can be prohibitively expensive. We adopt the implementation strategy from [4] to lower the computational costs of this process. Note that parameters like fbr (in SCutFBR.1), a and b (in our smoothing-based methods) only affect the thresholding step (i.e., step 5 of Algorithm 1) after $\phi^{(k)}$ had been trained. In [4], they reuse the same CV split for each value of fbr. We apply the same strategy to select a and b. This way, the $V+1$ models are reused for each (a, b). Only the threshold optimization is done for each parameter. Since most of the time is spent on training the linear models instead of threshold tuning, reusing the trained models saves considerable time (roughly m times faster). This implementation strategy is applied to all the methods above for our experiments.

4.3 Micro-F as Smoothed F-measure

Note that the methods proposed in Section 4.2 all require two levels of CV. Even with the technique speeding up the inner level of cross-validation, two levels of V-folds CV still require training $(V + 1)^2$ models for each label[2], which is time-consuming. In this section, we propose a heuristic that takes advantage of smoothed F-measure while being free from extra hyperparameters. This way, only one level of CV is required for each label.

We can notice that when optimizing the threshold on the current binary subproblem for label j, the micro-F calculated from label 1 to j acts as a smoothed F-measure:

$$\mathrm{F}_{1,j}^{\text{micro}} = \frac{\overbrace{\sum_{i=1}^{j-1} 2\mathrm{tp}_i}^{a} + 2\mathrm{tp}_j}{\underbrace{\sum_{i=1}^{j-1} (2\ \mathrm{tp}_i + \mathrm{fp}_i + \mathrm{fn}_i)}_{b} + (2\ \mathrm{tp}_j + \mathrm{fp}_j + \mathrm{fn}_j)}. \tag{8}$$

However, if we sequentially optimize (8) for each label, a and b would grow with the number of trained labels. When j is large, (8) does not serve as a nice surrogate for F-measure anymore. To benefit from the smoothing effect of micro-F while still optimizing for macro-F, we can consider optimizing the sum of micro-F and macro-F. In other words, micro-F can act as a regularizer that smooths the optimization of macro-F. Therefore, this setting should be less prone to overfitting than only optimizing macro-F. In detail, when tuning the threshold of the jth model, we optimize the following measure by treating the results of all previous labels as constants:

$$\begin{aligned} &\mathrm{F}_{1,j}^{\text{micro}}(T) + \mathrm{F}_{1,j}^{\text{macro}}(T) \qquad (9)\\ =&\frac{\sum_{i=1}^{j-1} 2\mathrm{tp}_i + 2\mathrm{tp}_j(T)}{\sum_{i=1}^{j-1} (2\ \mathrm{tp}_i + \mathrm{fp}_i + \mathrm{fn}_i) + (2\ \mathrm{tp}_j(T) + \mathrm{fp}_j(T) + \mathrm{fn}_j(T))}\\ &+\frac{1}{j}\left(\sum_{i=1}^{j-1} \frac{2\mathrm{tp}_i}{2\mathrm{tp}_i + \mathrm{fp}_i + \mathrm{fn}_i} + \frac{2\mathrm{tp}_j(T)}{2\mathrm{tp}_j(T) + \mathrm{fp}_j(T) + \mathrm{fn}_j(T)}\right). \end{aligned}$$

While we motivated the use of (9) from the viewpoint of smoothing the optimization of macro-F, an additional advantage is that reasonable micro-F may be obtained simultaneously. Since a single measure (macro-F) may not capture all aspects of a classifier and over-optimizing macro-F can lower the performance of micro-F, it can be beneficial to balance the optimization of two measures.

Based on the aforementioned idea, we propose the method:

micromacro – For each label j, apply Algorithm 1 using (9) as the measure.

Because the function (9) involves quantities associated with the thresholds of labels 1 to $j - 1$, a naive calculation would require looping over all previous labels. In Supplementary B, we provide some implementation details for evaluating (9) efficiently.

Since the order of labels may affect the performance of micro-macro. In Supplementary D, we provide experimental results to discuss the possible impact of label order on its performance.

5 EXPERIMENTS AND ANALYSES

5.1 Experimental Settings

5.1.1 Datasets. We compare the algorithms on six datasets. **RCV1-topics** [12], **EUR-Lex** [16] and **Wiki10-31K** [33] are text classifications tasks, each with different (in orders of magnitude) number of labels and different distribution of label frequency. We choose node classification for the remaining three datasets to see how our methods generalize to a different domain. They are **PPI** [7], **Flickr** [27] and **BlogCatalog** [27]. A major distinction between these two domains is that text data consists of high-dimensional, sparse tf-idf features, while node classification data consists of low-dimensional, dense, learned representations. The details about source of data

Table 1: Data statistics. For the column "distribution", we report the proportion of rare, medium and frequent labels in the mentioned order.

Dataset	#train	#test	#feature	#label	distribution
PPI	43,966	10,992	128	121	0/0/100
Flickr	64,410	16,103	128	195	0/12.3/87.7
BlogCatalog	8,249	2,063	128	39	2.56/25.6/71.8
RCV1-topics	23,149	781,265	47,236	101	4.95/27.7/67.3
EUR-Lex	15,449	3,865	186,104	3,956	63.5/31.5/4.95
Wiki10-31K	14,146	6,616	104,374	30,938	88.6/10.3/1.06

[2]Because we re-train with all data after cross-validation in both levels, the total models trained is $(V + 1)^2$ instead of V^2.

and preprocessing are in Supplementary C. We list the statistics for each dataset in Table 1, which includes the proportion of rare, medium and frequent labels in each dataset.

5.1.2 Methods. The comparison will include SCutFBR.n from Section 3.4, the two smoothing-based method from Section 4.2, and the micromacro method from Section 4.3. Besides, we also test the following algorithms:

- Binary relevance (BR) without threshold tuning.
- SCut [29]: For each label, this method applies Algorithm 1 using F-measure as the metric.
- SCutFBR (SCutFBR.1 in [4, 12]): For each label, this method performs a two-level CV. The outer level selects an fbr value with the best F-measure. The inner level applies Algorithm 1 with F-measure and FBR heuristic to select a threshold.
- Cost-sensitive [12, 13]: For each label, a one-level CV is performed to select C^+, the multiplier of the loss value for positive samples, with the best F-measure. Since this method optimizes the F-measure of each label, it is equivalent to optimizing macro-F.
- Cost-sensitive-micro [13]: A CV is performed to select a cost C^+ that is used for training all labels. The cost is selected to achieve the best micro-F. This method is included to see how well the compared methods perform on micro-F.

The details of all the parameters searched in each algorithm are listed in Supplementary C.

5.2 Main Results

We run each algorithm on each dataset with five different random seeds and report the averaged micro-F and macro-F. The results are listed in Table 2.

5.2.1 SCut's Overfitting Is Mitigated. First, BR without threshold tuning performs poorly on all datasets. SCut effectively improves micro-F and macro-F when there are not many rare labels. The improvements can be observed on three node classification data and RCV1-topics. However, on datasets dominated by rare labels (i.e., Wiki10-31K and EUR-Lex), SCut overfits and greatly lowers micro-F, due to many false positives generated by a too-low threshold, as analyzed in Section 3.1 and Figure 1. On these two datasets, FBR and our methods successfully mitigate the overfitting by thresholding at the upper bound of the negative distribution. Therefore, the micro-F improves drastically, and macro-F also increases.

5.2.2 Optimizing Macro-F. Although FBR can mitigate the overfitting of SCut, it comes with a cost. A decrease in macro-F compared to SCut can be observed on Flickr, BlogCatalog and RCV1-topics, which have fewer rare labels and more medium-to-frequent labels. This decrease shows that FBR cannot handle medium-to-frequent labels well. Compared to SCutFBR, SCutFBR.n achieved better macro-F on many datasets, supporting our claim that the highest decision value of all validation samples can be too pessimistic, and the highest decision value of the negative samples is a better choice.

Although SCutFBR.n improved upon SCutFBR, it still obtained macro-F lower than SCut on some datasets. In contrast, the macro-F of smooth-all is always better than or competitive with SCut and SCutFBR across all datasets. We believe this is because smooth-all can find thresholds better (in terms of macro-F) than the upper bound of negative distribution when more positive samples are present. See Section 5.3 for more explanation.

Now we focus on the cost-sensitive method, which also optimizes macro-F. Although not the best, it performs decently on the node classification datasets. However, its macro-F on text data is poor. This result agrees with previous studies that discovered tuning C^+ for SVM is not effective on text data [1]. Compared to the cost-sensitive method, thresholding seems more general, as it can handle data from different domains.

5.2.3 Balancing Micro-F and Macro-F. In practice, there is often a trade-off between micro-F and macro-F, which can be observed in Table 2. The smooth-all method is good at macro-F but sometimes give lower micro-F, as the results on Flickr and BlogCatalog show. On the other hand, cost-sensitive-micro is good at micro-F but sometimes gives significantly lower macro-F, as the results on PPI and Flickr show.

Only comparing macro-F can be unfair to SCutFBR since it was proposed to "minimize the impact on micro-average F1 at the expense of slightly lowering macro-average F1" [29]. On the datasets Flickr and BlogCatalog, where macro-F is lowered by SCutFBR, micro-F is indeed improved in exchange. However, if we compare SCutFBR (or SCutFBR.n) with micromacro, which also aims to balance both measures, we can see that micromacro is better on almost all datasets. It seems that micromacro finds a sweet spot between two extremes, achieving decent performance on both measures. Sometimes the performance of micromacro is even competitive to methods focusing on one measure only. Another advantage of micromacro is that no hyperparameters to tune, so only a single-level CV is needed. Therefore, the micromacro method is a lightweight and stable strategy for threshold selection.

5.2.4 Smooth-Each Overfits in Parameter Selection. Interestingly, smooth-each seems overfitting on EUR-Lex and Wiki10-31K since a drastic drop in micro-F and occurred and its macro-F is also lower than smooth-all. We explain why that happens. When we run Algorithm 1 with different values of (a, b), it gives a threshold $T_{a,b}$ for each pair of (a, b). Then, selecting (a, b) in the outer level of cross-validation is equivalent to optimizing the F-measures on the validation set using the set of thresholds $\{T_{a,b}\}$. When the thresholds in $\{T_{a,b}\}$ contain low thresholds, we risk overfitting F-measures again. This example tells us that, when the label is rare, not only directly tuning the threshold based on F-measure overfits easily, but selecting parameters or models based on F-measure may also be risky. We conclude that the parameters a and b should be selected over several labels, as in the smooth-all approach, so that they do not overfit the F-measure of a single label.

5.3 Improvements for Labels of Different Rarity

Because macro-F is the average F-measure over the labels, we can understand how labels of different rarity contribute to the improvement in macro-F by viewing their respective improvement in F-measure. In Figure 3, we present the relation between p and the improvement in F-measure of our methods (smooth-all and SCutFBR.n) relative to SCutFBR. First, we can notice that, as expected, our methods perform similarly to SCutFBR for extremely-rare labels (#positve $\leq$ 5) and highly-frequent labels (#positives

Table 2: Results on the test set, averaged over five random seeds. We report the micro-F and macro-F for each dataset. For readability, we multiply the value by 100. In Supplementary C, we provide the standard deviations and the significance tests of the experiments.

	PPI		Flickr		BlogCatalog		RCV1-topics		EUR-Lex		Wiki10-31K	
method	micro	macro	micro	macro	micro	macro	micro	macro	micro	macro	micro	macro
BR	44.58	16.86	23.26	13.21	29.00	13.28	80.39	49.57	52.63	17.40	26.22	2.06
SCut	53.58	**48.35**	37.91	30.16	32.50	26.10	81.26	62.45	21.83	27.14	1.88	8.27
SCutFBR	53.54	**48.35**	38.27	29.13	38.67	24.85	81.27	61.66	56.38	27.45	32.59	12.30
cost-sensitive	52.91	48.15	37.43	29.33	33.46	25.18	80.97	56.00	57.64	22.33	32.19	2.96
cost-sensitive-micro	**56.14**	40.26	39.21	23.86	39.97	23.36	80.99	54.64	**58.28**	25.56	**33.16**	6.46
SCutFBR.n	53.52	48.29	38.33	29.27	38.50	25.22	81.28	62.00	56.58	28.02	32.69	12.50
smooth-each	53.58	48.34	35.09	30.45	27.02	25.97	81.17	61.90	38.55	28.13	4.24	11.97
smooth-all	53.56	**48.35**	37.77	30.56	32.15	**26.31**	81.31	62.38	56.87	28.67	31.24	**12.69**
micromacro	54.43	48.21	**39.37**	29.59	**40.06**	25.72	**81.39**	**62.71**	56.65	**28.73**	32.55	12.62

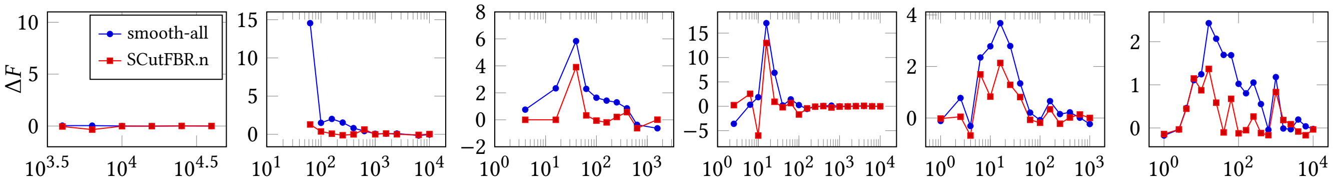

Figure 3: Improvement in F-measure (multiplied by 100) compared to SCutFBR. The y-axis shows the improvement in F-measure on the test set. The x-axis is the number of positive samples p for a label. Each dot represents a group of labels having similar numbers of positive samples. In each group, the improvement in F-measure for those labels is averaged and reported.

$\geq 10^3$). Second, on five of the six datasets, we can observe a peak of improvement for p around 10^1 to 10^2 (medium labels). The peak improvement can be big (over 10) on some datasets. This improvement in medium labels confirms our previous claim (in Figure 2) that FBR heuristic can make pessimistic decisions for medium labels, and our methods solve the problem. Moreover, we can observe that SCutFBR.n's improvement is smaller than that of smooth-all. This result implies that, although a reasonable choice, thresholding at the highest decision value of the negative distribution may not be the best when more positive samples are present.

Since our improvement is primarily in medium labels compared to FBR, the improvement would be invisible on datasets like PPI, which contains no medium labels. On the other hand, we can see larger improvements on datasets with more medium labels.

6 CONCLUSION

In this work, we focused on tuning the threshold for infrequent labels. This task has been observed to overfit easily and forms a challenging problem. First, we explained through examples that, due to how the F-measure is formulated, optimizing it when the positive samples are rare can easily lead to models that give a large number of false positives. Therefore, selecting thresholds or other hyperparameters based on F-measure can be dangerous on a rare label. Then, we explained why a previously proposed technique, the FBR heuristic, works well on rare labels. However, we also discovered that FBR heuristic could be too pessimistic in handling medium labels. To solve this issue, we first propose an improved variant of the FBR heuristic, which performs better and is more explainable. Then, we propose smoothing the F-measure when tuning the threshold. We derived a reasonable search range for the parameters in smoothed F-measure and theoretically proved that smoothing the F-measure can bring nice properties to the resulting threshold. Based on this idea, we also proposed jointly optimizing micro-F and macro-F as a lightweight alternative free from extra hyperparameters. Our proposed methods are empirically evaluated on text and node classification datasets. The results show that our methods consistently outperform previous approaches in two application domains.

ACKNOWLEDGMENTS

This work was supported by National Science and Technology Council of Taiwan grant 110-2221-E-002-115-MY3. The authors thank the reviewers for their constructive comments.

REFERENCES

[1] Janez Brank, Marko Grobelnik, Nataša Milić-Frayling, and Dunja Mladenić. 2003. *Training text classifiers with SVM on very few positive examples.* Technical Report. Technical Report MSR-TR-2003-34, Microsoft Corp.

[2] Wei-Cheng Chang, Daniel Jiang, Hsiang-Fu Yu, Choon-Hui Teo, Jiong Zhang, Kai Zhong, Kedarnath Kolluri, Qie Hu, Nikhil Shandilya, Vyacheslav Ievgrafov, Japinder Singh, and Inderjit S Dhillon. 2021. Extreme multi-label learning for semantic matching in product search. In *Proceedings of the 27th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD).*

[3] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. LIBLINEAR: a library for large linear classification. *Journal of Machine Learning Research* 9 (2008), 1871–1874. http://www.csie.ntu.edu.tw/~cjlin/papers/liblinear.pdf

[4] Rong-En Fan and Chih-Jen Lin. 2007. *A study on threshold selection for multi-label classification*. Technical Report. Department of Computer Science, National Taiwan University.

[5] Aditya Grover and Jure Leskovec. 2016. Node2vec: scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*. 855–864. https://doi.org/10.1145/2939672.2939754

[6] Haixiang Guo, Yijing Li, Jennifer Shang, Mingyun Gu, Yuanyue Huang, and Bing Gong. 2017. Learning from class-imbalanced data: Review of methods and applications. *Expert Systems With Applications* 73 (2017), 220–239.

[7] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*.

[8] Haibo He and Edwardo A. Garcia. 2009. Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering* 21, 9 (2009), 1263–1284.

[9] Kalina Jasinska, Krzysztof Dembczyński, Róbert Busa-Fekete, Karlson Pfannschmidt, Timo Klerx, and Eyke Hüllermeier. 2016. Extreme F-measure maximization using sparse probability estimates. In *Proceedings of The 33rd International Conference on Machine Learning (ICML)*. 1435–1444.

[10] Justin M. Johnson and Taghi M. Khoshgoftaar. 2019. Deep learning and thresholding with class-imbalanced big data. In *Proceedings of the 18th IEEE International Conference on Machine Learning and Applications (ICMLA)*. 755–762.

[11] Oluwasanmi O. Koyejo, Nagarajan Natarajan, Pradeep K. Ravikumar, and Inderjit S. Dhillon. 2015. Consistent Multilabel Classification. In *Advances in Neural Information Processing Systems*, Vol. 28.

[12] David D. Lewis, Yiming Yang, Tony G. Rose, and Fan Li. 2004. RCV1: A new benchmark collection for text categorization research. *Journal of Machine Learning Research* 5 (2004), 361–397.

[13] Li-Chung Lin, Cheng-Hung Liu, Chih-Ming Chen, Kai-Chin Hsu, I-Feng Wu, Ming-Feng Tsai, and Chih-Jen Lin. 2022. On the use of unrealistic predictions in hundreds of papers evaluating graph representations. In *Proceedings of the Thirty-Sixth AAAI Conference on Artificial Intelligence (AAAI)*. https://www.csie.ntu.edu.tw/~cjlin/papers/multilabel-embedding/multilabel_embedding.pdf

[14] Yu-Jen Lin. 2023. *On the Thresholding Strategies for Rare Labels in Multi-label Classification*. Master's thesis. Department of Computer Science and Information Engineering, National Taiwan University.

[15] Zachary C. Lipton, Charles Elkan, and Balakrishnan Naryanaswamy. 2014. Optimal thresholding of classifiers to maximize F1 measure. In *Proceedings of the Joint European Conference on Machine Learning and Knowledge Discovery in Databases (ECML/PKDD)*. 225–239.

[16] Johannes Loza Mencía, Eneldoand Fürnkranz. 2010. Efficient multilabel classification algorithms for large-scale problems in the legal domain. In *Semantic Processing of Legal Texts: Where the Language of Law Meets the Law of Language*, Enrico Francesconi, Simonetta Montemagni, Wim Peters, and Daniela Tiscornia (Eds.). Springer Berlin Heidelberg, 192–215.

[17] James Mullenbach, Sarah Wiegreffe, Jon Duke, Jimeng Sun, and Jacob Eisenstein. 2018. Explainable prediction of medical codes from clinical text. In *Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*. 1101–1111. https://doi.org/10.18653/v1/N18-1100

[18] Shameem A. Puthiya Parambath, Nicolas Usunier, and Yves Grandvalet. 2014. Optimizing F-measures by cost-sensitive classification. In *Advances in Neural Information Processing Systems*, Vol. 27.

[19] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. DeepWalk: Online Learning of Social Representations. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*. 701–710. https://doi.org/10.1145/2623330.2623732

[20] Ignazio Pillai, Giorgio Fumera, and Fabio Roli. 2013. Threshold optimisation for multi-label classifiers. *Pattern Recognition* 46, 7 (2013), 2055–2065.

[21] Foster Provost. 2000. Machine Learning from Imbalanced Data Sets 101. In *Proceedings of the AAAI Workshop on Imbalanced Data Sets*. 1–3.

[22] Erik Schultheis, Marek Wydmuch, Rohit Babbar, and Krzysztof Dembczynski. 2022. On missing labels, long-tails and propensities in extreme multi-label classification. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*. 1547–1557.

[23] Fabrizio Sebastiani. 2002. Machine learning in automated text categorization. *ACM Comput. Surv.* 34, 1 (2002), 1–47.

[24] Shai Shalev-Shwartz, Yoram Singer, and Nathan Srebro. 2007. Pegasos: primal estimated sub-gradient solver for SVM. In *Proceedings of the Twenty Fourth International Conference on Machine Learning (ICML)*.

[25] Aixin Sun, Ee-Peng Lim, and Ying Liu. 2009. On strategies for imbalanced text classification using SVM: A comparative study. *Decision Support Systems* 48, 1 (2009), 191–201.

[26] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. Line: Large-scale information network embedding. In *Proceedings of the 24th international Conference on World Wide Web (WWW)*. 1067–1077.

[27] Lei Tang and Huan Liu. 2009. Relational learning via latent social dimensions. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD)*. 817–826.

[28] Gang Wu and Edward Y. Chang. 2003. Class-Boundary Alignment for Imbalanced Dataset Learning. In *ICML Workshop on Learning from Imbalanced Data Sets II*. 49–56.

[29] Yiming Yang. 2001. A Study on Thresholding Strategies for Text Categorization. In *Proceedings of the 24th ACM International Conference on Research and Development in Information Retrieval*, W. Bruce Croft, David J. Harper, Donald H. Kraft, and Justin Zobel (Eds.). ACM Press, New York, US, New Orleans, US, 137–145.

[30] Hsiang-Fu Yu, Kai Zhong, Jiong Zhang, Wei-Cheng Chang, and Inderjit S. Dhillon. 2022. PECOS: Prediction for Enormous and Correlated Output Spaces. *Journal of Machine Learning Research* 23, 98 (2022), 1–32.

[31] Guo-Xun Yuan, Kai-Wei Chang, Cho-Jui Hsieh, and Chih-Jen Lin. 2010. A Comparison of Optimization Methods and software for Large-scale L1-regularized Linear Classification. *Journal of Machine Learning Research* 11 (2010), 3183–3234. http://www.csie.ntu.edu.tw/~cjlin/papers/l1.pdf

[32] Jiong Zhang, Wei-Cheng Chang, Hsiang-Fu Yu, and Inderjit S. Dhillon. 2021. Fast Multi-Resolution Transformer Fine-tuning for Extreme Multi-label Text Classification. 34 (2021), 7267–7280.

[33] Arkaitz Zubiaga. 2009. Enhancing Navigation on Wikipedia with Social Tags. In *Proceedings of Wikimania*.

Supplementary materials for "On the Thresholding Strategy for Infrequent Labels in Multi-label Classification"

Yu-Jen Lin
National Taiwan University
r10922a14@ntu.edu.tw

Chih-Jen Lin
National Taiwan University
cjlin@csie.ntu.edu.tw
Mohamed bin Zayed University of Artificial Intelligence
chihjen.lin@mbzuai.ac.ae

A PROOFS

A.1 Proof of Theorem 1

PROOF. We define $F^*(\text{tp}^*, \text{fp}^*, \text{fn}^*; a, b)$ to be the optimal value of smoothed F-measure attained by some threshold T^*, where tp^*, fp^* and fn^* are the corresponding counts generated by T^* on the set of samples. Then, we define $\hat{F}(\hat{\text{tp}}, \hat{\text{fp}}, \hat{\text{fn}}; a, b)$ to be the smoothed F-measure attained by any threshold higher than all scores, denoted as $\hat{T}$. Similarly, $\hat{\text{tp}}, \hat{\text{fp}}$ and $\hat{\text{fn}}$ denote the counts given by $\hat{T}$. Note that

$$\hat{\text{tp}} = 0, \hat{\text{fp}} = 0 \text{ and } \hat{\text{fn}} = \text{p}$$

because all samples are predicted as negative using the threshold $\hat{T}$.

Since the optimal value F^* must be larger or equal to $\hat{F}$, we can derive the following relation between tp^* and fp^*:

$$\begin{aligned} &\hat{F}(\hat{\text{tp}}, \hat{\text{fp}}, \hat{\text{fn}}; a, b) \le F^*(\text{tp}^*, \text{fp}^*, \text{fn}^*; a, b) \\ \implies & \frac{a+0}{b+2\cdot 0+0+\text{p}} \le \frac{a+2\text{tp}^*}{b+2\text{tp}^*+\text{fp}^*+(\text{p}-\text{tp}^*)} \\ \implies & a(b+\text{p}) + a(\text{tp}^*+\text{fp}^*) \le a(b+\text{p}) + 2\text{tp}^*(b+\text{p}) \\ \implies & \text{fp}^* \le \text{tp}^*\left(\frac{2(b+\text{p})}{a} - 1\right) \end{aligned} \tag{10}$$

□

A.2 Proof of Theorem 2

PROOF. Let i' be the index satisfying assumption (7). Then, let T' be a threshold immediately below $\phi(x_{i'})$ so that for all $i < i'$ we have $\phi(x_i) < T'$. For any threshold $T \ge T'$, we have $\text{fp}(T) = 0$ by the assumption in (7). Therefore, for all $T \ge T'$, the smoothed F-measure is

$$F(T; a, b) = \frac{a+2\text{tp}(T)}{b+\text{p}+\text{tp}(T)+\text{fp}(T)} = \frac{a+2\text{tp}(T)}{b+\text{p}+\text{tp}(T)}.$$

Taking the derivative of F with respect to tp, we have

$$\frac{dF}{d\text{tp}} = \frac{2(b+\text{p})-a}{(b+\text{p}+\text{tp})^2} > 0$$

because of the assumption that $a < 2(b+\text{p})$. This implies that as we increase T past the positive samples $i \ge i'$, tp decreases and the smoothed F-measure also decreases. Therefore, the optimal threshold T^* must be less than $\phi(x_{i'})$. □

A.3 Proof of Corollary 3

PROOF. If $a > 2(b+\text{p})$, then we have

$$\frac{2(b+\text{p})}{a} - 1 < 0.$$

If $\text{tp}^* > 0$, then fp^* must be negative according to Theorem 1, which is a contradiction since fp^* can only be non-negative. Therefore, $\text{tp}^* = 0$ and so $\text{fp}^* = 0$. That is, all samples are predicted as negative, so the threshold must be higher than all samples. □

A.4 Proof of Theorem 4

PROOF. Let

$$z = \max_{i, y_i=+1} \phi(x_i)$$

be the largest decision value of the positive samples. The statement of the theorem can be restated as

$$a < \frac{2(b+\text{p})}{N} \implies T^* \le z. \tag{11}$$

For any threshold $T \le z$, the corresponding smoothed F-measure $F(T; a, b)$ satisfies

$$F(T; a, b) = \frac{a+2\text{tp}(T)}{b+\text{p}+\text{tp}(T)+\text{fp}(T)} \ge \frac{a+2}{b+\text{p}+N} \tag{12}$$

because $\text{tp}(T) \ge 1$ and $\text{tp}(T)+\text{fp}(T) \le N$. For any $T > z$, the corresponding smoothed F-measure satisfies

$$F(T; a, b) = \frac{a+2\text{tp}(T)}{b+\text{p}+\text{tp}(T)+\text{fp}(T)} \le \frac{a}{b+\text{p}} \tag{13}$$

since $\text{tp}(T) = 0$ and $\text{fp}(T) \ge 0$. From (12) and (13), if we have

$$\frac{a+2}{b+\text{p}+N} > \frac{a}{b+\text{p}}, \tag{14}$$

then $T^* \le z$, our target in (11), must be true. Because inequality (14) is equivalent to

$$a < \frac{2(b+\text{p})}{N},$$

we have shown that (11) is true. □

B IMPLEMENTATION DETAILS OF MICROMACRO

In this supplementary section, we describe in detail how to efficiently evaluate (9) without looping over all the previous labels. In Algorithm 2, we give the detailed procedure for performing the micromacro method on one label. This algorithm is essentially Algorithm 1 with extensions for calculating micro-F, and we mark the difference between Algorithm 1 and Algorithm 2 in purple. The full micromacro algorithm simply calls Algorithm 2 for every label sequentially.

Below we give more details of Algorithm 2. When we are at step 5 for the k-th fold of the j-th label, we solve the following

Algorithm 2 The micromacro method for a label j. The differences from Algorithm 1 are in purple.

```
Input: A set of samples $D = \{(x_i, y_i) \mid y_i \in \{-1,+1\}\}$, the number of folds $V$, and the prediction counts $(\mathrm{tp}_{1,j-1}, \mathrm{fp}_{1,j-1}, \mathrm{fn}_{1,j-1})$.
Output: A trained scoring function $\phi$, a threshold $T$, and $(\mathrm{tp}_{1,j}, \mathrm{fp}_{1,j}, \mathrm{fn}_{1,j})$.
1:  Randomly partition $D$ to $V$ subsets $D_1, D_2, \dots, D_V$
2:  for $k = 1, 2, \dots, V$ do
3:      Train a model $\phi^{(k)}$ on $D \setminus D_k$
4:      On the validation set $D_k$, calculate the decision values $Z = \{\phi^{(k)}(x) \mid x \in D_k\}$
5:      Find the threshold $T^{(k)}$ maximizing $\frac{1}{j}\mathrm{F}_j(T^{(k)}) + \mathrm{F}^{\mathrm{micro}}_{1,j}(T^{(k)})$ on $Z$ and the counts $\mathrm{tp}^{(k)}, \mathrm{fp}^{(k)}$ and $\mathrm{fn}^{(k)}$ given by the optimal $T^{(k)}$
6:  end for
7:  $T = \frac{1}{V}\sum_{k=1}^{V} T^{(k)}$
8:  Train a $\phi$ on $D$
9:  $\mathrm{tp}_{1,j} = \mathrm{tp}_{1,j-1} + \sum_{k=1}^{V} \mathrm{tp}^{(k)}$
10: $\mathrm{fp}_{1,j} = \mathrm{fp}_{1,j-1} + \sum_{k=1}^{V} \mathrm{fp}^{(k)}$
11: $\mathrm{fn}_{1,j} = \mathrm{fn}_{1,j-1} + \sum_{k=1}^{V} \mathrm{fn}^{(k)}$
12: Return $(\phi, T)$
```

optimization problem on the decision values of the validation set D_k:

$$T^{(k)} = \arg\max_{T} \mathrm{F}^{\mathrm{macro}}_{1,j}(T) + \mathrm{F}^{\mathrm{micro}}_{1,j}(T).$$

Macro-F is the mean of F-measure of all currently trained binary problems:

$$\mathrm{F}^{\mathrm{macro}}_{1,j}(T) = \frac{1}{j}(\underbrace{\sum_{i=1}^{j-1} \mathrm{F}_i}_{\text{constant}} + \mathrm{F}_j(T)). \tag{15}$$

When training label j, the sum of F-measure F_i from label $i = 1$ to $j-1$ is an additive constant, which does not affect the result of optimization. Therefore, we can ignore the calculation for this constant. This trick allows macro-F to be evaluated efficiently on the current binary problem, disregarding j.

As for micro-F, the measure can be rewritten as follows:

$$\mathrm{F}^{\mathrm{micro}}_{1,j}(T) = \frac{2\,\mathrm{tp}_{1,j}(T)}{2\,\mathrm{tp}_{1,j}(T) + \mathrm{fp}_{1,j}(T) + \mathrm{fn}_{1,j}(T)}$$
$$= \frac{2\mathrm{tp}_{1,j-1} + 2\mathrm{tp}_j(T)}{(2\,\mathrm{tp}_{1,j-1} + \mathrm{fp}_{1,j-1} + \mathrm{fn}_{1,j-1}) + (2\,\mathrm{tp}_j(T) + \mathrm{fp}_j(T) + \mathrm{fn}_j(T))}. \tag{16}$$

The term $\mathrm{tp}_j(T)$ is simply the number of true positives given by thresholding at T on the validation set D_k. The same goes for false positives and false negatives. However, the terms like $\mathrm{tp}_{1,j-1}$ are more complex. They denote the sum of counts calculated from label 1 to $j-1$. What should we use to calculate these sums?

The first option is, for each label i from 1 to $j-1$, use its final model (the Φ and T given in steps 7 and 8 of Algorithm 1) to calculate the $\mathrm{tp}_i, \mathrm{fp}_i$ and fn_i on the whole training set. Then, sum these counts to get $\mathrm{tp}_{1,j-1}$ and so forth. However, this option requires an additional prediction on all training samples after training the final model at step 8 and risks being too optimistic. When the positive samples are rare, they can often be separated by the trained model, but there might be a lot of positive samples that are not seen in the training set and are predicted to be negative in the testing stage. Therefore, the counts estimated in the training set may be too optimistic. To mitigate this issue, we adopt a second option that utilizes the cross-validation results. At step 5 of Algorithm 2, when we obtained the optimal threshold ($T^{(k)}$ at step 5 of Algorithm 1) for the k-th fold, we also record the corresponding $\mathrm{tp}^{(k)}, \mathrm{fp}^{(k)}$ and $\mathrm{fn}^{(k)}$ achieved by $(\Phi^{(k)}, T^{(k)})$ on that validation set D_k. These counts are already calculated in Algorithm 1 since it goes through all the candidate thresholds and uses these counts to calculate F-measure. Therefore, we can obtain these counts with little extra effort. Then, the sum of these counts from all validation folds can be used as the counts for label j. In this approach, all training samples are utilized, but no additional prediction is required. The estimated counts would avoid being overly optimistic because the model $\Phi^{(k)}$ is not trained on the validation set D_k.

To evaluate $\mathrm{tp}_{1,j-1}, \mathrm{fp}_{1,j-1}$ and $\mathrm{fn}_{1,j-1}$ efficiently according to the mentioned strategy, we will maintain these accumulated sums over the training of each label. Whenever we finished training one label j, we add the counts for label j to the maintained accumulated sums, getting $\mathrm{tp}_{1,j}$ from $\mathrm{tp}_{1,j-1}$ (similarly for fp and fn). These updates are described in step 9 to 11 in Algorithm 2. The accumulated counts can then be directly used for training the next label without recalculation. This trick allows micro-F also to be evaluated efficiently without looping over the labels.

C DETAILS OF MAIN EXPERIMENTS

C.1 Datasets

For all six datasets, we use the preprocessed version available on the LIBSVM Data[1] repository. For the text classification data the tf-idf features provided on the website are used. The result on the standard testing subset is reported. For the node classification data, different features generated from different representation learning techniques are available. We adopt the representations learned with DeepWalk [19], Node2vec [5] and LINE [26], respectively for PPI, Flickr, and BlogCatalog. Since no standard testing subset is available for these datasets, we randomly split each of them into training (80%) and testing (20%) sets. The model is trained on the training set, and the result on the testing set is reported.

C.2 Parameters

C.2.1 Shared Settings. For all linear SVM, L2 regularization and L2 loss are used. The models are trained using the solver from LIBLINEAR [3]. In the main result, we use the option "-B 1" to include the bias term for all algorithms. Furthermore, the cost in cost-sensitive methods is passed as the "-w1" argument. Except for the mentioned parameters, other parameters (e.g., C for SVM) are not tuned and left as default.

For all algorithms' cross-validation (outer or inner level), we use $V = 3$ as the number of folds, and the split is stratified.

C.2.2 FBR Heuristic. For the search range of fbr in SCutFBR and SCutFBR.n, we follow the settings in [12]. That is, fbr is searched

[1]https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/multilabel.html

Table 3: Table of main results for node classification datasets, augmented with the standard deviations. The result is calculated from 5 random seeds.

	PPI		Flickr		BlogCatalog	
method	micro	macro	micro	macro	micro	macro
binary	44.58±0.0	16.86±0.0	23.26±0.0	13.21±0.0	29.0±0.0	13.28±0.0
SCut	53.58±0.04	48.35±0.01	37.91±0.13	30.16±0.07	32.5±1.88	26.1±0.49
SCutFBR	53.54±0.02	48.35±0.01	38.27±0.08	29.13±0.21	38.67±0.53	24.85±0.49
cost-sensitive	52.91±0.04	48.15±0.0	37.43±0.1	29.33±0.09	33.46±0.44	25.18±0.31
cost-sensitive-micro	56.14±0.0	40.26±0.0	39.21±0.0	23.86±0.0	39.97±0.0	23.36±0.0
SCutFBR.n	53.52±0.04	48.29±0.11	38.33±0.11	29.27±0.10	38.50±0.54	25.22±0.35
smooth-all	53.56±0.1	48.35±0.01	37.77±0.25	30.56±0.11	32.15±1.47	26.31±0.38
micromacro	54.43±0.01	48.21±0.01	39.37±0.11	29.59±0.17	40.06±0.23	25.72±0.41

Table 4: Table of main results for text classification datasets, augmented with the standard deviations. The result is calculated from 5 random seeds.

	RCV1-topics		EUR-Lex		Wiki10-31K	
method	micro	macro	micro	macro	micro	macro
binary	80.39±0.0	49.57±0.0	52.63±0.0	17.4±0.0	26.22±0.0	2.06±0.0
SCut	81.26±0.04	62.45±0.33	21.83±0.48	27.14±0.16	1.88±0.01	8.27±0.02
SCutFBR	81.27±0.02	61.66±0.36	56.38±0.21	27.45±0.17	32.59±0.06	12.3±0.03
cost-sensitive	80.97±0.02	56.0±0.27	57.64±0.08	22.33±0.09	32.19±0.01	2.96±0.01
cost-sensitive-micro	80.99±0.02	54.64±0.42	58.28±0.0	25.56±0.0	33.16±0.0	6.46±0.0
SCutFBR.n	81.28±0.03	62.00±0.06	56.58±0.11	28.02±0.11	32.69±0.05	12.50±0.03
smooth-all	81.31±0.07	62.38±0.21	56.87±0.25	28.67±0.19	31.24±0.52	12.69±0.07
micromacro	81.39±0.08	62.71±0.28	56.65±0.11	28.73±0.18	32.55±0.05	12.62±0.04

over

$$0.1, 0.2, \ldots, 0.8.$$

C.2.3 Cost-sensitive Methods. For the search grid of C^+ in cost-sensitive methods, we follow the settings in [13], so C^+ is searched over

$$\frac{2-t}{t} \text{ for } t = \frac{1}{7}, \frac{2}{7}, \ldots, 1.$$

C.2.4 Smoothed F-measure. Next, we provide the details of a and b searched in smooth-based methods. The implementation technique mentioned at the end of Section 4.2 allows us to search many pairs of a and b with little impact on training time. For both smooth-each and smooth-all, we search over these values of b:

$$0, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 1, 10, 10^{2}, 10^{3}, 10^{4}.$$

For each b in the list, we then select 30 values of a between the lower bound and the upper bound in the geometric series

$$\frac{2(b+\mathrm{p})}{N}, \frac{2(b+\mathrm{p})}{N}\cdot\Delta, \frac{2(b+\mathrm{p})}{N}\cdot\Delta^2, \ldots, 2(b+\mathrm{p}),$$

where $\Delta = \sqrt[29]{N}$ and N is the size of the training set. Notice that the bound for a depends on p, the number of positive samples for a label. When a and b are selected for a group of labels with different p (as in smooth-all), we use the smallest p in the group to determine the bound since smoothing is mainly for rare labels.

Table 5: Statistical significance test for the improvement of smooth-all over SCutFBR. Only the p-value (multiplied by 100 and rounded) for the improvement of macro-F is shown here.

PPI	Flickr	BlogCatalog	RCV1	EUR-Lex	Wiki10-31K
25.14	0.00	0.04	0.09	0.00	0.00

C.3 Detailed Experimental Results

C.3.1 Results with Standard Deviations. The results are in Table 3 and Table 4.

C.3.2 Statisfical Significance. We perform one-sided t-tests to test whether smooth-all and micromacro performed significantly better than SCutFBR. Since smooth-all specialize in macro-F, we only report the p-value for macro-F of each dataset in Table 5. The micromacro method and SCutFBR both try to balance micro-F and macro-F. Therefore, we calculate the p-value for both measures in Table 6. As these results show, the p-value is smaller than 0.05 in most cases. The reason why there is no significant improvement of macro-F on PPI is that it only has highly frequent labels, as explained in Section 5.3.

Table 6: Statistical significance test for the improvement of micromacro over SCutFBR. The p-value (multiplied by 100 and rounded) for the improvement in both measure is reported.

PPI		Flickr		BlogCatalog		RCV1-topics		EUR-Lex		Wiki10-31K	
micro	macro	micro	macro	micro	macro	micro	macro	micro	macro	micro	macro
0.0	100.0	0.0	0.3	0.03	0.75	0.35	0.01	0.89	0.0	82.59	0.0

Table 7: Comparison of performance with and without the bias term.

	RCV1-topics		EUR-Lex		Wiki10-31K	
	micro	macro	micro	macro	micro	macro
BR (no bias)	79.87	49.46	52.09	17.22	26.81	2.38
BR (with bias)	80.39	49.57	52.63	17.40	26.22	2.06
SCutFBR (no bias)	81.12	55.87	53.56	18.28	21.95	2.09
SCutFBR (with bias)	81.27	61.62	56.39	27.48	32.59	12.30
micromacro (no bias)	81.22	56.57	54.25	19.24	21.39	2.14
micromacro (with bias)	81.39	62.72	56.67	28.74	32.52	12.62

C.4 About the Bias Term of SVM

In Supplementary C, we mentioned that we include the bias term for all experiments using the option "-B 1" of LIBLINEAR. This choice is because we discovered that removing the bias term greatly affects the performance, especially for text datasets. We present a simple comparison in Table 7. The performance difference is present in both SCutFBR and our methods. This result differs from previous studies on text data that observed no difference using the bias term [24, 31]. There are several factors possibly contributing to this difference:

- The datasets considered in the mentioned studies are more balanced, and the number of positive samples is high compared to the rare labels considered in this study.
- In [31], they used accuracy as the performance measure, while we used micro-F and macro-F, which are more sensitive to rare labels.
- Threshold adjustion was not used in these studies. So the benefit brought by the bias term (i.e., a better orientation of the learned hyperplane) is not fully utilized. We can observe the results of BR in Table 7. The performance difference between no bias and with bias is small if the threshold is not adjusted.

D LABEL ORDER FOR MICROMACRO

For the micromacro method, the order of labels may affect the performance. In this section, we provide experimental results to discuss the impact of label order on the performance. We experiment with the following four variants of micromacro:

- micromacro: This is the default variant used for the main experiments. We go through the labels in the order they appear in the file of training samples without other processing.
- micromacro-freq: The labels are sorted in descending order according to the number of positive samples p. We start from the most frequent label and end with the rarest.
- micromacro-tail: The labels are sorted in ascending order according to the number of positive samples p. We start from the rarest label and end with the most frequent.
- micromacro-rand: The labels are randomly shuffled.

The results in Table 8 and Table 9 show that the default, freq and rand variants perform similarly. On the other hand, the tail variant sometimes obtains significantly lower micro-F. A possible reason is that the micro-F does not yet have the smoothing effect when tuning the threshold of the first few labels. That is, the a in (8) is still zero or too small. Therefore, when applying micromacro, the first few labels should not all be rare. Although the random variant may also have rare labels as its first few labels, the probability does not seem high enough to have a big impact. As for the default variant, because the label that appeared early are often frequent, it also does not suffer much from the problem of rare labels. In conclusion, the default, freq, and rand variants seem more robust than the tail variant.

E OCCURENCES OF FIGURE 1 AND FIGURE 2

E.1 Occurences of Figure 1

In Section 3.1, we mentioned that when performing CV on rare labels, positive validation samples are often mixed with the negative examples (Figure 1). This behavior is why directly optimizing F-measure overfits badly. To show that this situation indeed occurs in practice, we perform additional experiments on EUR-Lex and Wiki10-31K, which have a large proportion of rare labels. For each data, we perform a 3-fold CV over the rare labels ($p < 10$) and count the situations like Figure 1. Precisely, for each validation set, we check the following conditions:

(1) The largest decision value of positive samples, say z, is less than -0.5.
(2) The number of negative samples with decision values higher than z is larger than 100.

The results in Table 10 show that the situation indeed occurs on many validation sets and affects a lot of rare labels. This observation

Table 8: Comparison of micromacro with different label orders on node classification datasets. The average and standard deviation calculated from 5 random seeds are reported.

	PPI		Flickr		BlogCatalog	
method	micro	macro	micro	macro	micro	macro
micromacro	**54.43**±0.01	48.21±0.01	**39.37**±0.11	29.59±0.17	**40.06**±0.23	25.72±0.41
micromacro-freq	54.13±0.03	48.24±0.02	39.17±0.15	**29.71**±0.21	39.52±0.38	25.64±0.32
micromacro-rand	54.41±0.06	48.13±0.04	39.22±0.03	29.12±0.19	39.16±0.81	25.03±0.25
micromacro-tail	53.22±0.05	**48.29**±0.02	39.11±0.1	29.22±0.07	31.24±3.23	**26.12**±0.55

Table 9: Comparison of micromacro with different label orders on text classification datasets. The average and standard deviation calculated from 5 random seeds are reported.

	RCV1-topics		EUR-Lex		Wiki10-31K	
method	micro	macro	micro	macro	micro	macro
micromacro	81.39±0.08	62.71±0.28	56.65±0.11	28.73±0.18	32.55±0.05	12.62±0.04
micromacro-freq	81.33±0.01	62.53±0.28	56.68±0.03	**28.85**±0.16	32.00±0.05	12.62±0.05
micromacro-rand	**81.41**±0.05	62.48±0.33	**56.81**±0.09	28.69±0.07	**32.68**±0.14	**12.66**±0.07
micromacro-tail	81.20±0.03	**62.75**±0.12	54.88±0.06	28.73±0.13	31.70±0.05	12.63±0.02

Table 10: Frequency of Figure 1 occurrences during cross-validation. Affected labels means the percentage of labels that has at least one validation set like Figure 1.

Dataset	Validation sets like Figure 1 (%)	Affected labels (%)
EUR-Lex	23.21	53.73
Wiki10-31K	36.70	69.54

Table 11: Frequency of Figure 2 occurrences during cross-validation.

Dataset	Validation sets like Figure 2 (%)	Affected labels (%)
Flickr	37.31	59.09
BlogCatalog	32.43	48.65
RCV1	1.75	2.63
EUR-Lex	4.45	10.82
Wiki10-31K	10.82	22.45

is consistent with the fact that SCut performs disastrously on these two datasets in Table 2.

E.2 Occurences of Figure 2

In Section 3.3, we mentioned that FBR can be too pessimistic in handling medium labels, giving up the easy positives (Figure 2). We performed a 3-fold CV over the medium-to-frequent labels ($10 \leq$ number of positive samples $p \leq 1000$) of each dataset to check if this situation occurs in practice with $fbr = 0.4$ (middle value of the search range used in [12]). Precisely, for each validation set, we check the following conditions:

(1) There are easy positives (positive samples that are above all negative samples) in the validation set.
(2) The FBR heuristic sets the threshold higher than the easy positives.

We exclude PPI from this result because it only has highly-frequent labels. The results in Table 11 show that situations like Figure 2 indeed occur in practice, with node classification datasets being influenced more heavily.