

國立臺灣大學電機資訊學院資訊工程學系

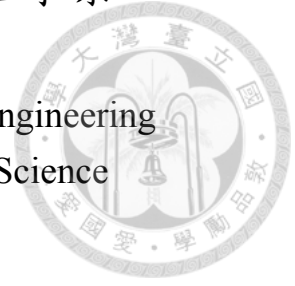
碩士論文

Department of Computer Science and Information Engineering

College of Electrical Engineering and Computer Science

National Taiwan University

Master Thesis



多標籤分類中對稀有標籤的閾值調整策略之討論

On the Thresholding Strategies for Rare Labels in

Multi-label Classification

林育任

Yu-Jen Lin

指導教授：林智仁博士

Advisor: Chih-Jen Lin, Ph.D.

中華民國 112 年 6 月

June, 2023

國立臺灣大學碩士學位論文
口試委員會審定書

MASTER' S THESIS ACCEPTANCE CERTIFICATE
NATIONAL TAIWAN UNIVERSITY



多標籤分類中對稀有標籤的閾值調整策略之討論

On the Thresholding Strategies for Rare Labels in
Multi-label Classification

本論文係林育任君（學號 R10922A14）在國立臺灣大學資訊工程學系人工智慧碩士班完成之碩士學位論文，於民國 112 年 6 月 30 日承下列考試委員審查通過及口試及格，特此證明。

The undersigned, appointed by the Master Program of Artificial Intelligence offered by the Department of Computer Science and Information Engineering on 30 June 2023 have examined a Master' s thesis entitled above presented by LIN, YU-JEN (student ID: R10922A14) candidate and hereby certify that it is worthy of acceptance.

口試委員 Oral examination committee:

林智仁

(指導教授 Advisor)

蔡銘暉

林軒田

系主任/所長 Director:

洪士濼



Acknowledgements

I would like to thank my advisor, Professor Chih-Jen Lin, for his guidance and support, which is important for completing this work. I would also like to thank the people at lab 528 for all the help offered during my graduate study, both on the research and career decisions. Finally, I am grateful to my family for their unconditional support.



摘要

在多標籤分類任務中，標籤出現次數間的不平衡是個常見的問題。對於出現次數稀少的標籤來說，用來產生二元預測值的預設閾值往往不是最佳的。然而，在過去的文獻中已觀察到直接透過最佳化 F 值來選取新閾值容易造成過擬合。在此篇論文中，我們解釋了為什麼藉由調整閾值來最佳化 F 值以及類似的評價指標時特別容易過擬合。接下來，我們分析了 FBR 啟發法——一個既有的對於此過擬合的解法。我們為其成功之處提供了解釋，但也點出 FBR 的潛在問題。針對所發現的問題，我們提出了一個新技巧，在閾值最佳化時對 F 值做平滑化處理。我們以理論證明，如果選取了恰當的參數，平滑化可為調整後的閾值帶來良好的性質。延續平滑化的概念，我們更進一步提出同時最佳化微觀平均 F 與巨觀平均 F 的方法。其享有平滑化所帶來的好處，但是更為輕量化，不需要調整額外的超參數。我們在文字與節點分類的資料集上驗證了新的方法的有效性，其一致的超越了 FBR 啟發法。

關鍵字： 多標籤分類，F 值，稀有標籤，文本分類，節點分類，支持向量機，閾值調整



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 for F-measure has been observed to overfit easily. In this work, we explain why tuning the thresholds for rare labels to optimize F-measure (and similar metrics) is particularly prone to overfitting. Then, we analyze the FBR heuristic, a previous technique proposed to address the overfitting issue. We explain its success but also point out its potential problems. Then, 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.

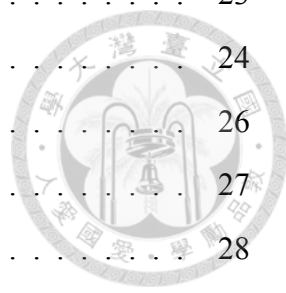
Keywords: multi-label classification, F-measure, rare labels, text classification, node classification, support vector machines, threshold adjustment



Contents

口試委員會審定書	i
Acknowledgements	ii
摘要	iii
Abstract	iv
1 Introduction	1
2 Preliminaries	4
2.1 Problem Setting	4
2.2 Evaluation Metrics	5
2.3 Previous Works	6
3 Why FBR Works and Its Issues	9
3.1 An Interesting Behavior of FBR	9
3.2 The Benefit of Lowering the Threshold	11
3.3 Issues of FBR Heuristic	13
3.4 A new variant of SCutFBR	14
4 Smoothed F-measure	15
4.1 Selecting a and b	16
4.2 Comparing Smoothed F-measure with the FBR Heuristic	18
4.3 Micro-F as Smoothed F-measure	21
5 Experiments and Analyses	23

5.1	Experimental Settings	23
5.2	Main Results	24
5.3	Trade-off between micro-F and macro-F	26
5.4	Occurences of Figure 3.1 and Figure 3.2	27
5.5	Improvements for labels of different rarity	28
5.6	Discussion on selecting a and b	29
5.7	Effect of the bias term of SVM	32
6	Conclusion	34
	Bibliography	35
A	Proofs	39
A.1	Proof of Theorem 1	39
A.2	Proof of Corollary 2	40
A.3	Proof of Theorem 3	40
A.4	Proof of Theorem 4	41
A.5	Proof of Theorem 5	43
B	Implementation details of micromacro	44
C	Details of main experiments	46
D	Details of Auxiliary Experiments	49
D.1	Occurences of Figure 3.2	49
D.2	Occurences of Figure 3.1	49





List of Figures

3.1	Example of a bad threshold maximizing F-measure.	10
3.2	Example of FBR making a pessimistic decision.	11
5.1	Improvement in F-measure compared to SCutFBR.	27
5.2	Contour plot of macro-F obtained through cross-validation with respect to parameters a and b	30
5.3	Contour plot of macro-F with respect to parameters a and b , calculated for label groups of different frequency.	31



List of Tables

5.1	Data statistics.	24
5.2	Main experimental results.	25
5.3	Frequency of Figure 3.1 occurrences during cross-validation.	27
5.4	Frequency of Figure 3.2 occurrences during cross-validation.	28
5.5	Comparison of performance with and without the bias term.	32
C.1	Main results on node classification with standard deviations.	47
C.2	Main results on text classification with standard deviations.	47



List of Algorithms

2.1	SCut (one label)	6
-----	----------------------------	---



Chapter 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 [21], product search or recommendation [2], medical code prediction [15], and node classification [17, 5, 24].

Among many of these 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 labels correctly. In such situations, metrics like Precision@k or nDCG@k are common choices [28, 30]. However, there are still many important applications with a smaller number of labels (e.g., $\leq 10,000$), such as node classification [17, 5, 24] or medical code prediction [15]. In these cases, we aim to predict all the relevant labels for an instance instead of only the top ones. In this case, 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, our main focus in this work. Among the two metrics mentioned above, macro-F is known to take infrequent labels into account more effectively [13, 20]. 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 handle imbalanced datasets. Common techniques include over/under-sampling [8, 6], cost-sensitive methods [8, 6], and threshold adjustment [10, 23]. 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 [27]. However, applying SCut to infrequent labels has been observed to overfit easily, a situation that leads to a challenging research problem [27, 9]. 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 [27] is the only technique to battle this issue. Despite FBR's success in the past [11, 4], it is a heuristic. The reason why it performed well is not well-understood. Therefore, in this work, we aim to analyze the difficulties of tuning thresholds for infrequent labels and provide a more principled solution to the problems discovered.

In previous works on thresholding, the classifier for each label is usually a linear classifier [11, 4]. 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 point out 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 [27], 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 with the FBR heuristic.

- To solve the problems discovered, 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 many theoretical justifications.
- Both FBR and our methods using smoothed F-measure 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 Chapter 2, we formally define the problem and review some previous works. In Chapter 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. In Chapter 4, we introduce the smoothed F-measure for regularizing the tuning of thresholds and propose several algorithms based on this idea. Finally, we present experimental results in Chapter 5 and conclude this study in Chapter 6.



Chapter 2

Preliminaries

In this chapter, we first introduce the setting of multi-label classification. Then, the evaluation metrics micro-F and macro-F are introduced. Finally, we give a review of previous methods.

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 j th 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 $\Phi(x) : X \rightarrow \{-1, +1\}^L$ using the training samples, such that it correctly predicts the set of relevant labels given a new x unseen in the training set.

The model Φ can be decomposed into L components:

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

Each component Φ_j is then trained to solve the subproblem of whether a sample is associated with label j , which is a binary classification problem. The components Φ_j are often implemented by a scoring function $\phi_j : X \rightarrow \mathbb{R}$ outputting a decision value for each label

j . Then, a threshold T is applied to $\phi_j(x)$ to generate binarized predictions:

$$\Phi_j(x) = \begin{cases} +1 & \phi_j(x) > T \\ -1 & \phi_j(x) \leq T \end{cases} \quad (2.1)$$



When Φ_j and ϕ_j are solved independently for each j as a binary classification problem, this approach is called the binary relevance method. 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 $T = 0$. Nevertheless, the proposed method should apply to other models, as long as a score is assigned to each label and thresholded as in (2.1). This includes 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

$$F = \frac{2tp}{2tp + fp + fn}, \quad (2.2)$$

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

When there are multiple labels, 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

Algorithm 2.1 SCut (one label)

Input: A set of samples $D = \{(x_i, y_i) \mid y_i \in \{-1, +1\} \text{ for } i = 1, 2, \dots, N\}$, a metric $f(T)$, number of folds V

Output: A trained scoring function ϕ , a threshold T

- 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 $T^{(k)}$ that maximizes $f(T^{(k)})$ on Z .
 - 6: **end for**
 - 7: $T = \frac{1}{V} \sum_{k=1}^V T^{(k)}$
 - 8: Train a ϕ on D
 - 9: **Return** (ϕ, T)
-



counts calculated only using label j . Macro-F and micro-F can be expressed as:

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

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. When the dataset is imbalanced, and a different evaluation metric is adopted, not tuning the threshold can be a “critical mistake” [19]. Several works have demonstrated the importance of threshold tuning for traditional machine learning models [27, 4, 1, 23] and neural networks [10].

SCut – Instead of using the default threshold T in (2.1), SCut [27] 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 2.1. 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 2.1. For simplicity, we assume the decision values $\phi(x_i)$ 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 $i = 1, \dots, N - 1$. Then the optimization of the measure f is done by searching through these thresholds:

$$(T_0, T_1, \dots, T_N) \text{ where}$$

$$T_i = \begin{cases} \phi^{(k)}(x_1) - \epsilon & i = 0 \\ \frac{1}{2} (\phi^{(k)}(x_i) + \phi^{(k)}(x_{i+1})) & 0 < i < N \\ \phi^{(k)}(x_N) + \epsilon & i = N \end{cases}$$

for some small constant $\epsilon > 0$

When our goal is to optimize macro-F, it suffices to apply Algorithm 2.1 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 [27]. 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, 18] (but a single pass over the labels may be enough to produce a good model). In each iteration of the cyclic optimization, 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 for label j .

The form of SCut given in Algorithm 2.1 assumes that the score functions ϕ_j can be independently trained as in the binary relevance method. To apply the idea to neural networks where the output unit for each label is trained together, we can modify Algorithm 2.1

so that all output units are trained together on the training set in step 3. However, their threshold is tuned and averaged separately on the validation set in step 5 and step 7, respectively.

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 [16]. In the case of SVMs, for each binary subproblem, the weight C^+ for the loss of positive samples is selected via CV [1, 11, 12]. 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 [23, 11, 1].

FBR heuristic – While SCut is a reasonable approach, it has been discovered to overfit easily on rare labels [27, 9]. 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 [27]. If the maximal F-measure on the validation set does not exceed a prespecified value fbr at step 5 of Algorithm 2.1, 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 [27], was shown effective [11, 4]. In addition, the author proposed a less-used variant of the method, called SCutFBR.0 [27], 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 of 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 why it is successful but also point out its potential problems.



Chapter 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 Chapter 4.

3.1 An Interesting Behavior of FBR

A previous study pointed out that the FBR heuristic shows an interesting behavior of lowering the threshold for rare labels, which is associated with improving macro-F [4]. This behavior occurs when the validation set consists of only negative 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. In this case, the FBR heuristic sets the threshold to the highest decision value of the negative distribution, which is often lower than the original threshold at 0.

In addition to this situation pointed out by [4], we further discovered that lowering thresholds can still happen even if there are positive instances in the validation set. When the number of positive training samples is too few to represent the full positive distribution, the positive samples in the validation set are usually predicted as negative [1]. We give

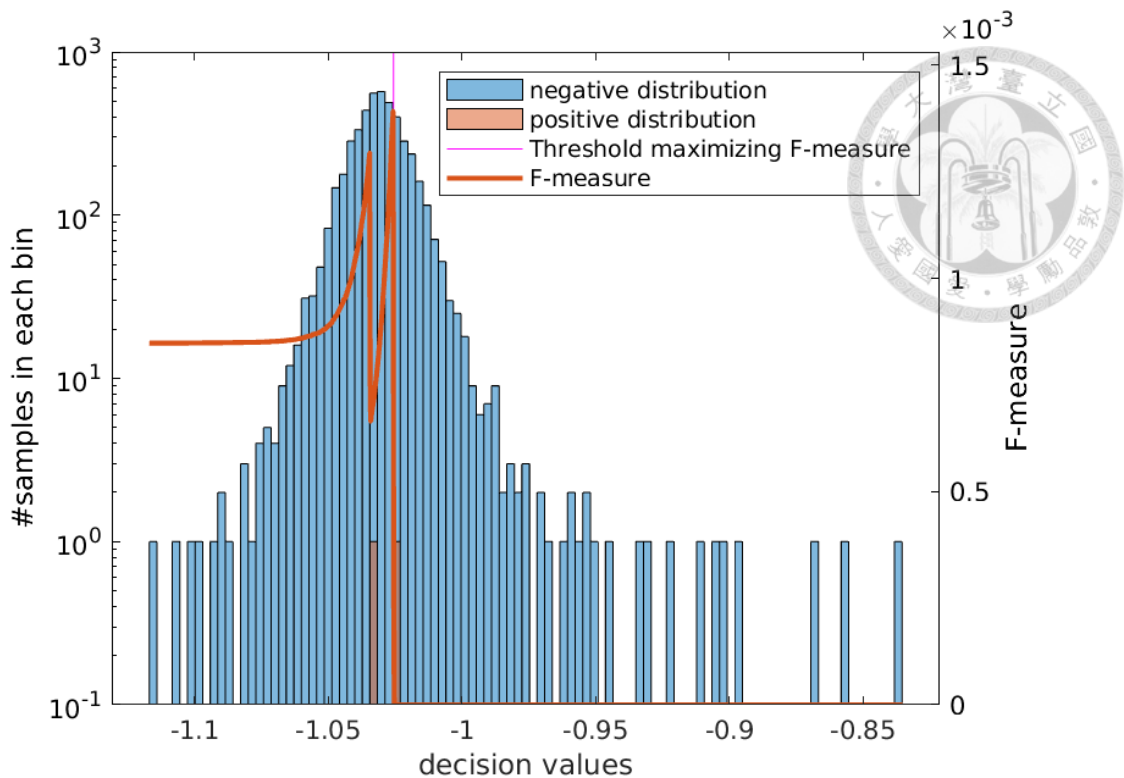


Figure 3.1: The distribution of the decision values from one of the validation sets of a binary subproblem from Wiki10-31K. In this case, the optimal value of the F-measure is clearly a bad threshold. The threshold is so low that it generates many false positives. Also, notice that the gain in F-measure is almost zero (1.4×10^{-3} in this case).

an illustration in Figure 3.1. In this case, blindly optimizing the F-measure on decision values of validation data leads to a too-low threshold that gives many false positives. This behavior can be understood from the definition of F-measure in Equation (2.2). When $tp = 0$, the 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 Section 5.4, we perform experiments to show that situations like Figure 3.1 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, avoiding many false positives. Notice that the resulting threshold is still much lower than the original threshold 0.

Nevertheless, SCut overfits in the first place because F-measure does not penalize false positives when $tp = 0$. We conclude that although F-measure is widely used to evaluate

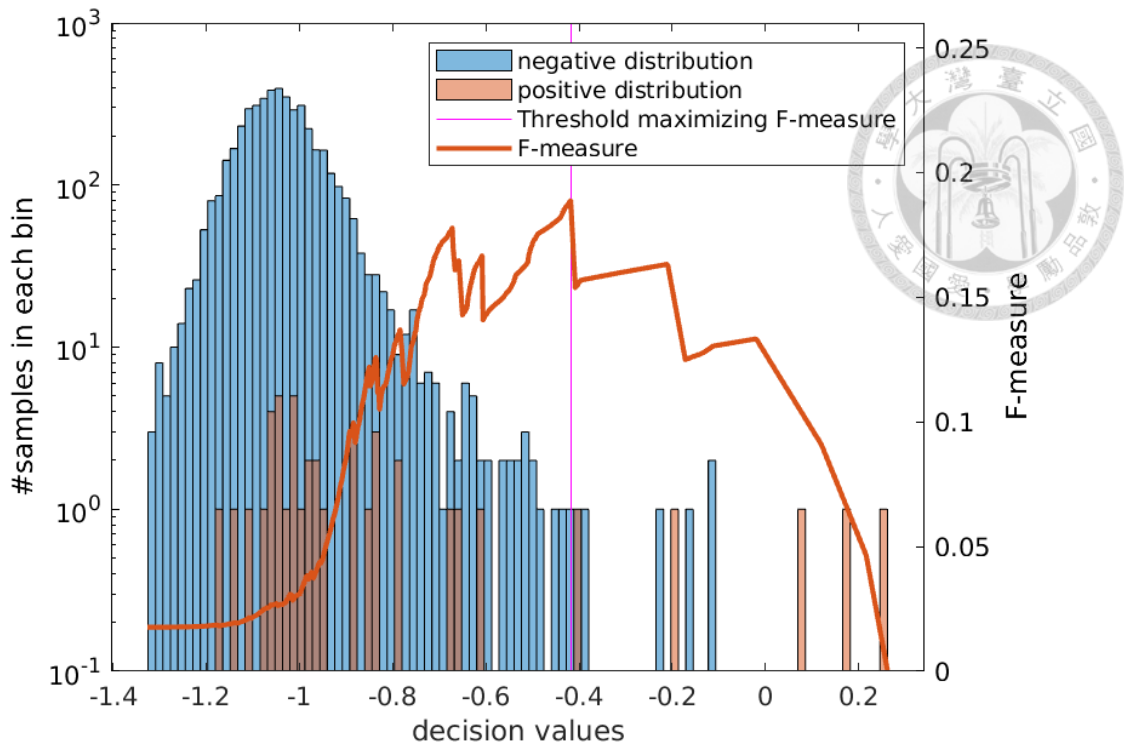


Figure 3.2: The distribution of the decision values from one of the validation sets of a binary subproblem from Wiki10-31K. The plotted F-measure is calculated by setting the threshold to the mid-point between adjacent decision values and counting the corresponding tp, fp and fn.

a model’s performance on rare labels, directly optimizing it with threshold selection is inappropriate. In Chapter 4, we propose effective methods to fix this issue.

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 Lowering the Threshold

We have seen that FBR has an interesting behavior of lowering the threshold for rare categories, which was likely unaware by the original author [27]. In this section, we explain why this is the desired behavior when the positive samples are rare.

In imbalanced scenarios, it has been known that the decision boundaries learned by common algorithms are usually skewed toward the minority, which results in under-predicting the minority class. This behavior occurs because the positive samples are too few to spec-

ify the positive distribution boundary fully. A synthetic example for SVMs demonstrating this behavior was given in [26]. Furthermore, [1, 23] tested SVMs on real-world text data and discovered that SVM's default threshold is too high, misclassifying many positive samples as negative in the testing phase. In recent years, similar problems have also been discovered on neural networks, showing that the default 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 $p(y = +1 | w^T x + b)$ 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 while not affecting the performance of frequent labels, we provide a solution in Chapter 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 3.1), 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 in practice. In this section, we point out some issues of this heuristic. Then, we give our solutions to these issues in Chapter 4.

1. It is unclear how to choose the fbr value, and no obvious default values exist. To deal with this, the original authors added an extra level of CV to select the parameter fbr [11]. However, two levels of CV lengthen the training time.
2. 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 Chapter 1). We give an illustration in Figure 3.2. In this case, we can see that there are three positive samples on the right, and the F-measure is never higher than 0.2 for any threshold. We call these positive samples not below any negative samples the “**easy positives**”. 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 gives no false positives on the validation set while not giving up the easy positives. Assuming the testing distribution is not far from the validation distribution, we can reasonably expect the latter strategy not to give many false positives while successfully detecting some true positives in the testing phase. We argue that a reasonable threshold should not give up the easy positives.

Situations like Figure 3.2 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 lower F-measure in the validation set. In Section 5.4, we experimentally show that situations like Figure 3.2 indeed occur in practice.

3.4 A new variant of SCutFBR

In Section 3.2, we argued that the reason why FBR works well is that it coincidentally sets the threshold to the upper bound of the negative distribution when dealing with rare labels, which we argued to be a reasonable strategy. Furthermore, in Section 3.3, we pointed out that setting the threshold to the highest decision value can be too pessimistic when the highest decision values are positive. To verify our claim that setting the threshold to the upper bound of the negative distribution is better, we propose and experiment with a new variant of the FBR heuristic:

SCutFBR.n: After optimizing F-measure with threshold selection, if the attained F-measure is less than the parameter fbr , the threshold is set to the **highest decision value among the negative samples**. A two-level cross-validation is used, where the outer level selects the parameter fbr and the inner-level selects the threshold.

Although setting the threshold to the highest decision value of the negative distribution is reasonable when the positive samples are very few, it can still be rather naive when a label have more positive samples, which allows better threshold to be found. The method we propose in the next section will allow these better threshold to be found when more positive samples are present, while falling back to the highest decision of the negative distribution when the positive samples are very few.



Chapter 4

Smoothed F-measure

In Section 3.1, we explained that optimizing F-measures results in a too-low threshold since F-measure does not penalize false positives when $tp = 0$. We propose the following solution that smooths the F-measure by introducing some constants a and b where $a > 0$ and $b \geq 0$:

$$F(tp, fp, fn; a, b) = \frac{a + 2tp}{b + 2tp + fp + fn} \quad (4.1)$$

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 $tp = fp = fn = 0$. 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 $tp(T)$ for the number of true positives (similarly for fp and fn) and $F(T; a, b)$ for the smoothed F-measure. To demonstrate how smoothing regularizes threshold tuning, 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 $D = \{(\phi(x_i), y_i) \mid y_i \in \{-1, +1\} \text{ for } i = 1, 2, \dots, N\}$ be given. We assume $\phi(x_i) \neq \phi(x_j)$ for all $i \neq j$. Furthermore, let a and b be given such that $a > 0$ and $b + p > 0$, 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

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

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



The proof is in Appendix A.1. Theorem 1 shows that when optimizing the smoothed F-measure, the resulting threshold never gives too many false positives unless a proportional number of true positives 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.

By running Algorithm 2.1 with the smoothed F-measure, threshold tuning can be regularized given proper values of a and b .

4.1 Selecting a and b

To choose the proper a and b , one basic approach is to have a list of candidate values

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

Then, we can perform CV to select the value with the best result. Notice that we do not use a list of values for a and b separately and search over all combinations because the proper range of a depends on b , as we will see in this section, where we derive range of reasonable values of a to search over.

For the upper bound of a , we use the following corollary of Theorem 1.

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

The proof is in Appendix A.2. Clearly, unconditionally placing the threshold higher than all decision values is too pessimistic, giving up the easy positives, which we argued

to be problematic in Section 3.3. Therefore, $2(b + p)$ can be an upper bound for a since any value larger gives an undesirable behavior.

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



Theorem 3. Under Assumption 1, if a and b also satisfy

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

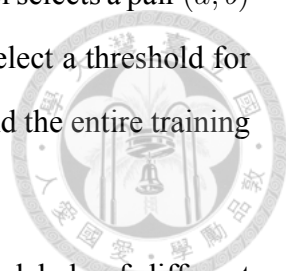
then the threshold T^* maximizing the smoothed F-measure is always less than or equal to the largest decision value among the positive samples.

The proof is in Appendix A.3. Recall that the original motivation of smoothing is to prevent lowering the threshold to always include at least one positive. Theorem 3 shows that if a is too small, the behavior of including at least one positive sample would always occur. Therefore, $2(b + p)/N$ 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 go through a prespecified geometric sequence for b . Then, for each b , we search through a geometric sequence for the value of a within the range derived in Corollary 2 and Theorem 3. 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:

1. **smooth-each:** For each label j , perform a two-level cross-validation. The outer level selects a pair of (a, b) . The inner level selects the threshold using Algorithm 2.1 with $F(T; a, b)$ as the metric. Using the best (a^*, b^*) , a final model for label j is produced by running Algorithm 2.1 with all training data. In short, a and b are selected independently for each label.

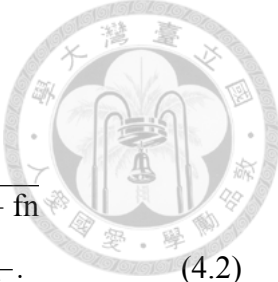
- 
2. **smooth-all**: Perform a two-level cross-validation. The first level selects a pair (a, b) for all labels. The second level uses the same given (a, b) to select a threshold for each label using Algorithm 2.1. Then, using the best (a^*, b^*) and the entire training set, run Algorithm 2.1 again for each label.
 3. **smooth-group**: Intuitively, the best (a, b) may vary between labels of different frequencies. Therefore, this approach separates labels into groups with a similar number of positive samples and applies the smooth-all approach to each group.

The above procedures all involve a CV process for searching a, b . Under each training/validation split in the first level of CV, and some given (a, b) , a call to Algorithm 2.1 requires 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 2.1 for each pair searched, then $m(V + 1)$ models in total have to be 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 calculation of the threshold (i.e., step 5 of Algorithm 2.1) after the scoring function 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 running 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 recommended, and we apply it to all the methods above for our experiments.

4.2 Comparing Smoothed F-measure with the FBR Heuristic

In this section, we discuss the similarities and differences between smoothed F-measure and the FBR heuristic. To gain more insight into smoothed F-measure, we can rewrite it

as

$$\begin{aligned}
 F(\text{tp}, \text{fp}, \text{fn}; a, b) &= \frac{a + 2\text{tp}}{b + 2\text{tp} + \text{fp} + \text{fn}} \\
 &= \frac{a}{b + 2\text{tp} + \text{fp} + \text{fn}} + \frac{2\text{tp}}{b + 2\text{tp} + \text{fp} + \text{fn}} \\
 &= \frac{a}{b + \text{p} + \text{tp} + \text{fp}} + \frac{2\text{tp}}{b + 2\text{tp} + \text{fp} + \text{fn}}. \tag{4.2}
 \end{aligned}$$


The third equality is due to the fact that $\text{p} = \text{tp} + \text{fn}$. Because $\text{tp} + \text{fp}$ is the number of samples predicted as positive (i.e., the number of samples above the threshold), the denominator

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

is nondecreasing as the threshold moves down. Therefore, a higher threshold leads to a larger value of

$$\frac{a}{b + \text{p} + \text{tp} + \text{fp}}.$$

Also, the second term in (4.2) resembles the original F-measure (when $b = 0$, it becomes F-measure). From these observations, we can interpret smoothed F-measure as regularizing the threshold towards higher values. With this in mind, we formally prove that optimizing smoothed F-measure can exhibit a similar behavior to FBR.

Theorem 4. Given $fbr \in (0, 1]$ and a set of distinct decision values along with their binary labels $\{(\phi(x_i), y_i) \mid y_i \in \{-1, +1\} \text{ for } i = 1, 2, \dots, N\}$. There exist parameters a and b such that the results of optimizing the corresponding smoothed F-measure have the following property: If the original F-measure does not reach the value fbr , then the threshold is set higher than the highest decision value.

The proof is in Appendix A.4. Theorem 4 roughly shows that our method generalizes FBR. Therefore, when a and b are thoroughly searched, our method should not perform worse than using FBR.

In Section 3.2, we argued that setting the threshold to an estimated upper bound of the negative distribution is desired when the label is rare, and FBR turned out to have this behavior. We now explain how smoothing naturally achieves this. When Algorithm 2.1

performs cross-validations on rare labels, two cases often 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 samples in the training set. The positive samples in the validations set are mixed in the negative distribution like Figure 3.1.



In the case of no positive samples in the validation set, $tp = fn = 0$ and lowering the threshold only gives more false positives. If we have $a, b > 0$, the smoothed F-measure reaches the maximum when the threshold is higher than all decision values. Therefore, optimizing smoothed F-measure sets the threshold to the upper bound of negatives. Note that we also require $b > 0$ here to prevent dividing by zero when the threshold is higher than all decision values. In the second case, where the positive samples are all predicted as negative in the validation set, the achieved F-measures are usually low, like Figure 3.1. Given proper a and b , the threshold would be set to the highest decision value, as indicated by Theorem 4.

In Section 3.3, we argue that FBR can be too pessimistic when dealing with medium labels. It might give up the easy positives simply because the F-measure is not high enough. We now show that smoothed F-measure does not give up the easy positives.

Theorem 5. Under Assumption 1, we further assume that the decision values are sorted in ascending order and $a < 2(b + p)$. If there exists i' such that

$$y_i = +1 \text{ for all } i \geq i', \quad (4.3)$$

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

The proof is in Appendix A.5. Theorem 5 shows that if a is lower than the upper bound derived in Corollary 2, then the threshold optimizing the smoothed F-measure will be lower than every easy positive sample $\phi(x_{i'})$.

Summarizing this section, we showed that smoothing possesses the desirable properties of FBR but does not make pessimistic decisions like FBR.

4.3 Micro-F as Smoothed F-measure

Note that the methods proposed in Section 4.1 all require two levels of CV for selecting a and b . Even though in Section 4.1 we described a technique to speed up the inner level of cross-validation, two levels of V -folds CV would still require training $(V + 1)^2$ models in total for each label¹, which is time-consuming. In this section, we propose another 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:

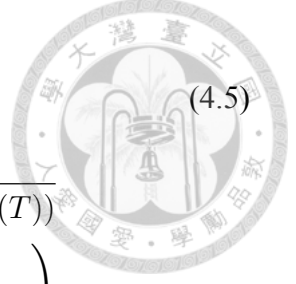
$$F_{1,j}^{\text{micro}} = \frac{\sum_{i=1}^{j-1} 2\text{tp}_i + 2\text{tp}_j}{\underbrace{\sum_{i=1}^{j-1} (2\text{tp}_i + \text{fp}_i + \text{fn}_i) + (2\text{tp}_j + \text{fp}_j + \text{fn}_j)}_b} \quad (4.4)$$

However, if we sequentially optimize (4.4) for each label, a and b would grow with the number of trained labels. When j is large, (4.4) 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 j th model, we optimize the following measure by treating the

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

results of all previous labels as constants:

$$\begin{aligned}
 & F_{1,j}^{\text{micro}}(T) + F_{1,j}^{\text{macro}}(T) \\
 &= \frac{\sum_{i=1}^{j-1} 2\text{tp}_i + 2\text{tp}_j(T)}{\sum_{i=1}^{j-1} (2\text{tp}_i + \text{fp}_i + \text{fn}_i) + (2\text{tp}_j(T) + \text{fp}_j(T) + \text{fn}_j(T))} \\
 &+ \frac{1}{j} \left(\sum_{i=1}^{j-1} \frac{2\text{tp}_i}{2\text{tp}_i + \text{fp}_i + \text{fn}_i} + \frac{2\text{tp}_j(T)}{2\text{tp}_j(T) + \text{fp}_j(T) + \text{fn}_j(T)} \right)
 \end{aligned} \tag{4.5}$$



While we motivated the use of (4.5) 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 2.1 using (4.5) as the measure.

Because the function (4.5) involves quantities associated with the thresholds of labels 1 to $j - 1$, a naive calculation would have a complexity of $O(L)$, which is higher than the $O(1)$ cost of evaluating the smoothed F-measure (4.1). In Appendix B, we provide some implementation details for evaluating (4.5) efficiently.

With this new method, the disadvantages mentioned in Chapter 3 disappeared. The outer level of CV for selecting hyperparameters is not required, making training faster. The thresholds that are too low are avoided by the micro-F term instead of FBR heuristic since micro-F serves as a smoothed F-measure.



Chapter 5

Experiments and Analyses

5.1 Experimental Settings

Datasets. We compare the algorithms on six datasets. **RCV1-topics** [11], **EUR-Lex** [14] and **Wiki10-31K** [31] 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** [25] and **BlogCatalog** [25]. 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 and preprocessing are in Appendix C. We list the statistics for each dataset in Table 5.1, which includes the proportion of rare, medium and frequent labels in each dataset.

Methods. The comparison will include SCutFBR.n from Section 3.4, the three smoothing-based method from Section 4.1, and the micromacro method from Section 4.3. Besides, we also test the following algorithms:

- Binary relevance (BR) without threshold tuning.
- SCut [27]: Apply Algorithm 2.1 to every label with F-measure.
- SCutFBR (SCutFBR.1 in [11, 4]): For each label, perform a two-level CV. The outer level selects a fbr value with the best F-measure. The inner level applies

Table 5.1: 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

Algorithm 2.1 with F-measure and FBR heuristic to select a threshold.

- Cost-sensitive [11, 12]: 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 [12]: 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 micromacro method performs on micro-F.

The details of all the parameters searched in each algorithm are listed in Appendix C. Our code for experiments will be made publicly available after the review process.

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 5.2. First, we can see that binary relevance 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, which is due to many false positives generated by a too-low threshold, as analyzed in Section 3.1 and Figure 3.1. On these two datasets, FBR and our methods

Table 5.2: 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 Appendix C, we provide the standard variations of the experiments.

method	PPI		Flickr		BlogCatalog		RCV1-topics		EUR-Lex		Wiki10-31K	
	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
smooth-group	53.57	48.35	37.51	30.59	32.20	26.18	81.27	62.41	56.69	28.59	30.80	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

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.

However, FBR 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 means that FBR cannot handle medium-to-frequent labels well. Compared to SCutFBR, SCutFBR.n achieved better macro-F on 5 out of 6 datasets and better micro-F on 4 out of 6 datasets. This improvement supports 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 lower macro-F than SCut on some datasets. In contrast, the macro-F of smooth-all and smooth-group is always better or competitive as that of SCut and SCutFBR across all datasets. Therefore, these two methods can find thresholds even better than the upper bound of negative distribution and thus are better at optimizing macro-F.

Interestingly, smooth-each seems overfitting on EUR-Lex and Wiki10-31K since a drastic drop in micro-F occurred. We explain why that happens. When we run Algorithm 2.1 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. We conclude that the parameters a and b should be selected over several labels, as in smooth-all or smooth-group.

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” [27]. 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 for a balance in both measures, we can see that micromacro is better on almost all datasets and measures.

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. In contrast, it performs poorly on text data. 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.

5.3 Trade-off between 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 5.2. The smooth-all and smooth-group methods are good at optimizing macro-F but sometimes give lower micro-F, as the results on Flickr and BlogCatalog show. On the other hand, cost-sensitive-micro is pretty good at optimizing micro-F but sometimes gives significantly lower macro-F, as the results on PPI and Flickr show. Among the methods tested, micromacro found a sweet spot between these two extremes, achieving decent performance on both measures. Sometimes the micro-F achieved is even better than that of cost-sensitive-micro, which specializes in micro-F. Moreover, there are also cases when its macro-F is higher than methods targeting macro-F. Another advantage of micromacro is that no hyperparameters have to be tuned, so only a single-level CV is required. Therefore, the micromacro method is a lightweight and stable alternative for threshold selection.

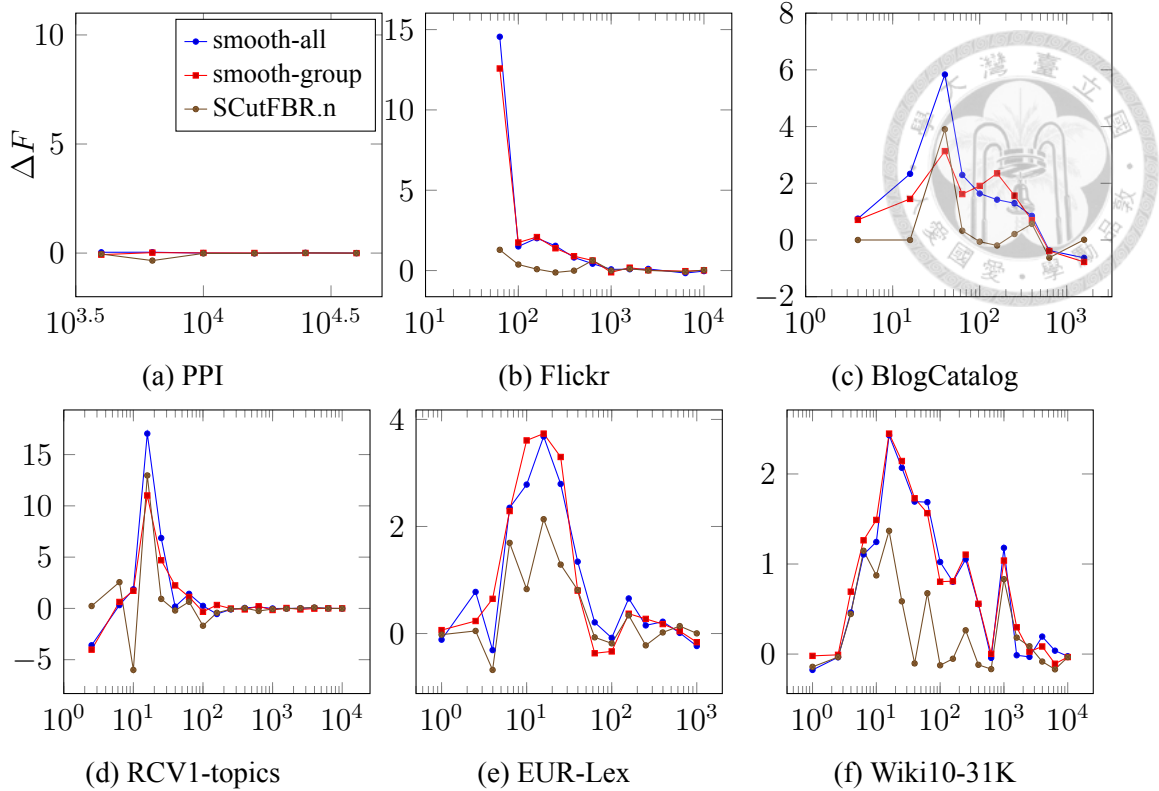


Figure 5.1: Improvement in F-measure (multiplied by 100) compared to SCutFBR. The y-axis shows the improvement in F-measure. The x-axis is the number of positive samples p for a label. Each dot represents a group of labels having similar numbers of positive samples. In each group, the improvement in F-measure for those labels is averaged and reported.

Table 5.3: Frequency of Figure 3.1 occurrences during cross-validation.

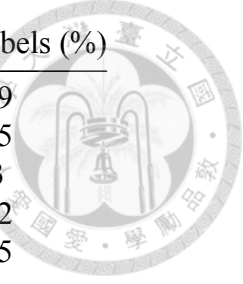
Dataset	Validation Sets Like Figure 3.1 (%)	Affected Labels (%)
EUR-Lex	23.21	53.73
Wiki10-31K	36.70	69.54

5.4 Occurences of Figure 3.1 and Figure 3.2

In Section 3.1, we mentioned that when performing CV on rare labels, positive validation samples are often mixed with the negative examples (Figure 3.1). 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 and count the situations like Figure 3.1. The results in Table 5.3 show that the situation indeed occurs on rare labels a lot. This result is consistent with the fact that SCut performs

Table 5.4: Frequency of Figure 3.2 occurrences during cross-validation.

Dataset	Validation Sets Like Figure 3.2 (%)	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



disastrously on these two datasets in Table 5.2.

In Section 3.3, we mentioned that FBR can be too pessimistic in handling medium labels, giving up easy positives when they can be better handled (Figure 3.2). 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 occurs in practice with $fbr = 0.4$ (middle value of the search range used in [11]). We exclude PPI from this result because it only has highly-frequent labels. The results in Table 5.4 show that situations like Figure 3.2 indeed occurs in practice, with node classification datasets being influenced more heavily. In the next section, we present more results demonstrating that this pessimistic behavior affects FBR’s performance on medium-to-frequent labels.

Details of these two experiments are given in Appendix D.

5.5 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 5.1, we present the relation between p and the improvement in F-measure of our methods (smooth-all, smooth-group, and SCutFBR.n) relative to SCutFBR. On five of the six datasets, we can observe a peak of improvement for p around 10^1 to 10^2 (medium labels). On the other hand, our methods perform similarly to SCutFBR for extremely-rare labels ($\#positve \leq 5$) and highly-frequent labels ($\#positives \geq 10^3$). The peak improvement can be big (over 10) on some datasets. This improvement in medium labels confirms our previous claim (in Figure 3.2) that FBR heuristic can make pessimistic decisions for medium labels, and smoothing solves the problem. Moreover,

we can observe that SCutFBR.n’s improvement is smaller than that of smoothing-based methods. This result confirms our previous claim in Section 3.4 that setting the threshold to the highest decision value of the negative distribution, although better than the original FBR, is still a strategy that can be further improved.

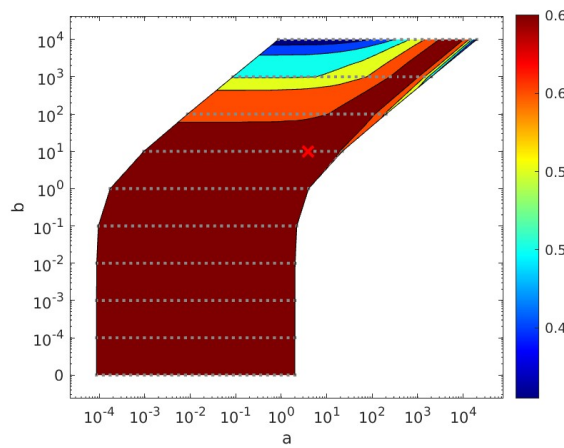
Since our improvement is primarily in medium labels compared to FBR, whether there is a significant improvement in macro-F depends on the distribution of label frequency. For datasets like PPI containing no medium labels, our methods’ improvement would be invisible. On the other hand, we can see larger improvements on datasets with more medium labels.

5.6 Discussion on selecting a and b

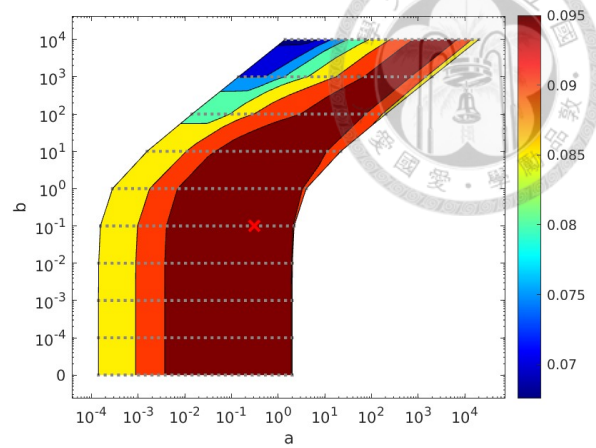
In this section, we evaluate the sensitivity of our methods with respect to the smoothing parameters and discuss the proper range for searching the parameter b . We also discuss whether labels of different number of positive samples need to select a and b separately.

In Figure 5.2, we plot the contour plot of macro-F obtained through cross-validation with respect to different combinations of a and b . The figure is drawn using the results from the smooth-all algorithm. As the figure shows, the validation performance is usually good for small values of b ($\leq 10^2$), and the best values of b are usually between 10^{-1} and 10^1 . As the value of b increases, the region of high macro-F narrows.

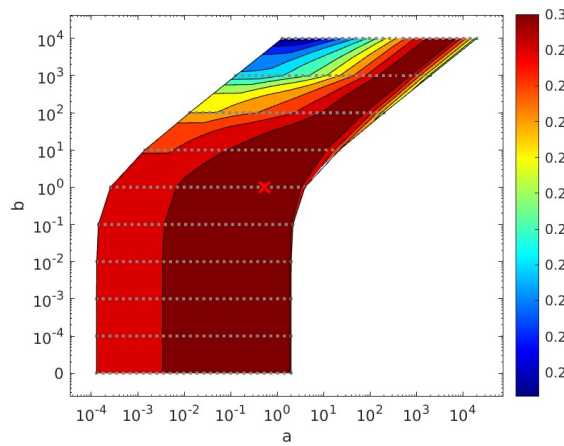
To examine the effect of smoothing for different labels, we group labels according to their number of positive samples and plot the contour of macro-F calculated on each group of labels in Figure 5.3. We use only the dataset Wiki10-31K since its variation in number of positive samples between labels is the largest. Other datasets also exhibit similar behavior. For rare labels ($10^0 \sim 10^1$), it seems that both large or small values of parameters work well. Interestingly, the best a and b follows a relation $b = a \cdot c + d$ for some $c, d > 0$, which is of the same shape of the upper and lower bound of a in log scale. For more frequent labels, small values of b works better and the region of high macro-F narrows as b increases. This narrowing happens when b becomes comparable with the number of positive samples.



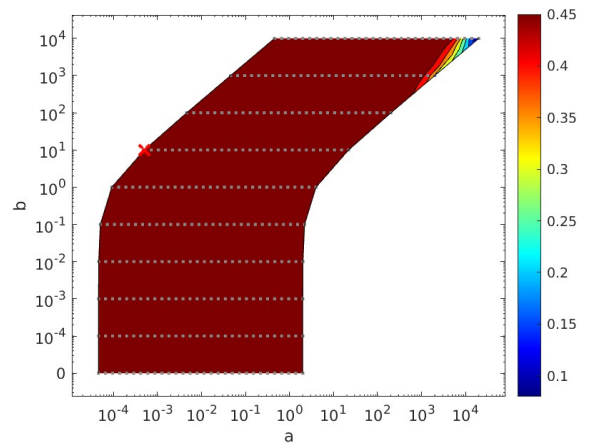
(a) rcv1



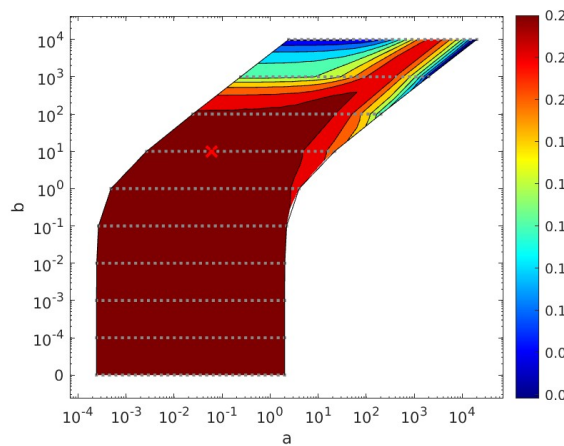
(b) Wiki10-31K



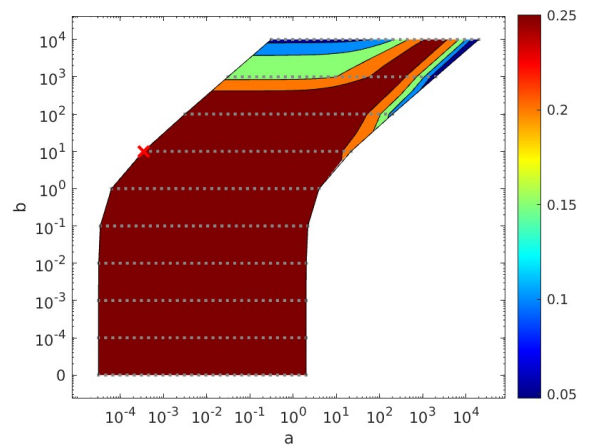
(c) EUR-Lex



(d) PPI



(e) BlogCatalog



(f) flickr

Figure 5.2: Contour plot of macro-F obtained through cross-validation with respect to parameters a and b . The parameters searched is marked in gray dots. The parameter that reached the highest value of macro-F is marked with a red cross.

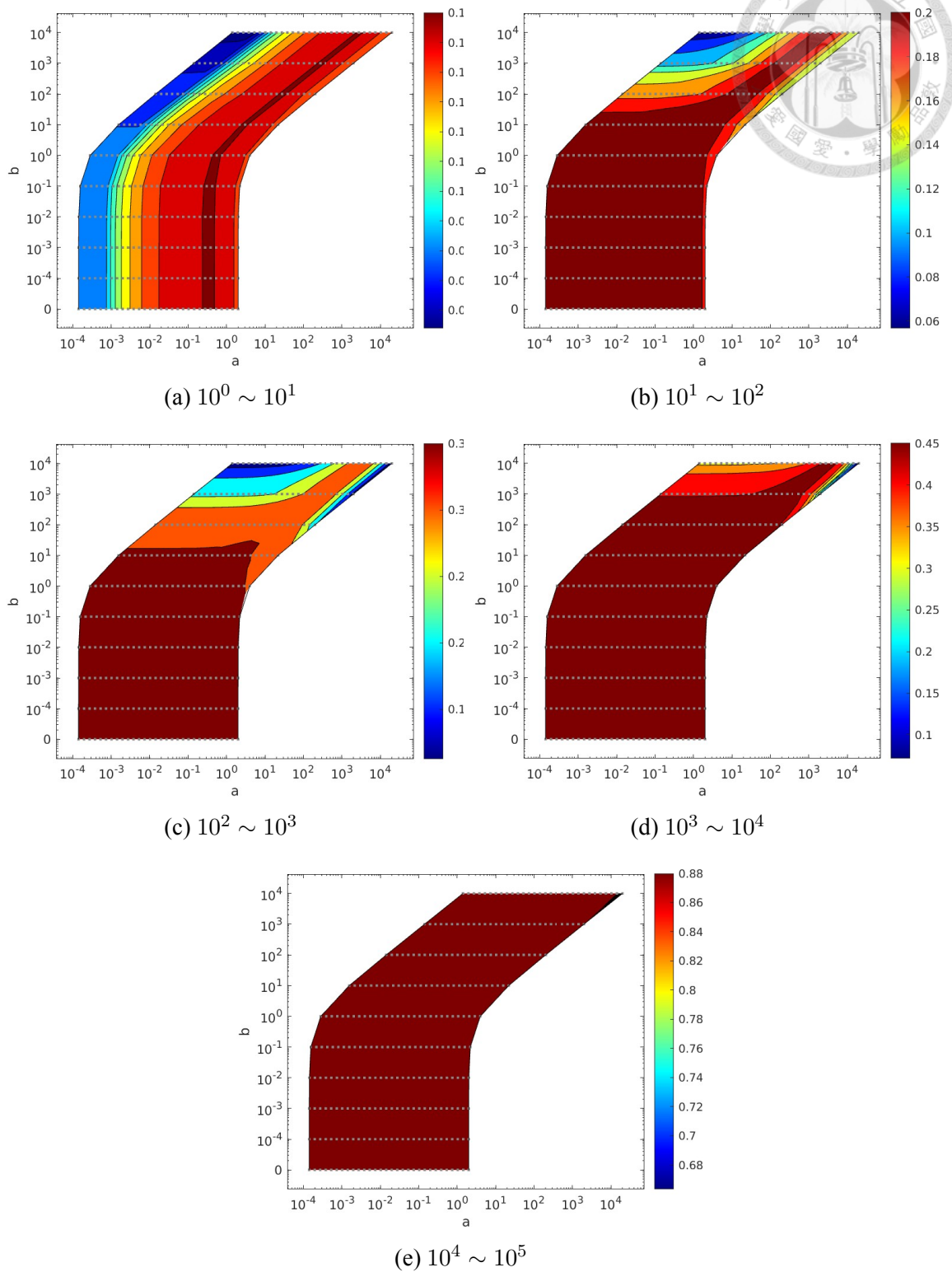


Figure 5.3: Contour plot of macro-F with respect to parameters a and b , calculated for label groups of different frequency. The parameters searched is marked in gray dots, while the upper and lower bound of parameter a is marked with a black line

Table 5.5: 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

Although in Figure 5.3 the contour for labels of different frequencies are different, the regions of high macro-F overlap. Therefore, selecting parameters a and b for all labels together or separately for each group should have similar performances. This conclusion is consistent with the results in Table 5.2 and Figure 5.1, where it is shown that smooth-all and smooth-group generally have similar performances.

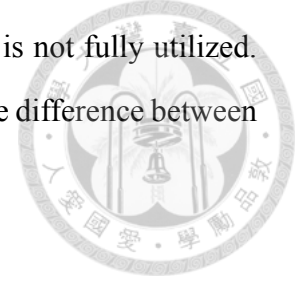
Judging from both Figure 5.2 and Figure 5.3, it seems that, instead of the range 10^4 adopted in Section C, searching up to 10^2 for the parameter b should be adequate in practice.

5.7 Effect of the bias term of SVM

In Appendix 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 5.5. 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 [22, 29]. There are several factors possibly contributing to this difference:

- The datasets considered were more balanced, and the number of positive samples is high compared to the rare labels considered in this study.
- In [29], they used accuracy as the performance measure, while we used micro-F and macro-F, which are more sensitive to rare labels.

- Threshold adjustment 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 5.5. The performance difference between no bias and with bias is small if the threshold is not adjusted.





Chapter 6

Conclusion

In this work, we focused on tuning the threshold for infrequent labels, which has been observed to overfit easily and forms a challenging problem. We explained through examples that this overfitting is largely due to how F-measure (and similar metrics) is formulated. Then, we explained why a previously proposed technique, the FBR technique, 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 proposed using smoothed F-measure as a surrogate 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.

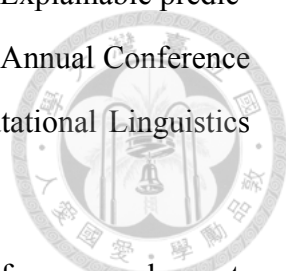
Our methods based on smoothed F-measures are empirically evaluated on text and node classification datasets. The results show that our methods consistently outperform previous approaches in two application domains.




Bibliography

- [1] J. Brank, M. Grobelnik, N. Milić-Frayling, and D. Mladenić. Training text classifiers with SVM on very few positive examples. Technical report, Technical Report MSR-TR-2003-34, Microsoft Corp, 2003.
- [2] W.-C. Chang, D. Jiang, H.-F. Yu, C.-H. Teo, J. Zhang, K. Zhong, K. Kolluri, Q. Hu, N. Shandilya, V. Ievgrafov, J. Singh, and I. S. Dhillon. 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), 2021.
- [3] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. LIBLINEAR: a library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, 2008.
- [4] R.-E. Fan and C.-J. Lin. A study on threshold selection for multi-label classification. Technical report, Department of Computer Science, National Taiwan University, 2007.
- [5] A. Grover and J. Leskovec. Node2vec: Scalable feature learning for networks. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), page 855–864, 2016.
- [6] H. Guo, Y. Li, J. Shang, M. Gu, Y. Huang, and B. Gong. Learning from class-imbalanced data: Review of methods and applications. *Expert Systems With Applications*, 73:220–239, 2017.

- [7] W. Hamilton, Z. Ying, and J. Leskovec. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*, 2017.
- [8] H. He and E. A. Garcia. Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering*, 21(9):1263–1284, 2009.
- [9] K. Jasinska, K. Dembczyński, R. Busa-Fekete, K. Pfannschmidt, T. Klerx, and E. Hüllermeier. Extreme f-measure maximization using sparse probability estimates. In *Proceedings of The 33rd International Conference on Machine Learning (ICML)*, pages 1435–1444, 2016.
- [10] J. M. Johnson and T. M. Khoshgoftaar. Deep learning and thresholding with class-imbalanced big data. In *Proceedings of the 18th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 755–762, 2019.
- [11] D. D. Lewis, Y. Yang, T. G. Rose, and F. Li. RCV1: A new benchmark collection for text categorization research. *Journal of Machine Learning Research*, 5:361–397, 2004.
- [12] L.-C. Lin, C.-H. Liu, C.-M. Chen, K.-C. Hsu, I.-F. Wu, M.-F. Tsai, and C.-J. Lin. 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)*, 2022.
- [13] Z. C. Lipton, C. Elkan, and B. Naryanaswamy. 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)*, pages 225–239, 2014.
- [14] J. Loza Mencía, Eneldoand Fürnkranz. Efficient multilabel classification algorithms for large-scale problems in the legal domain. In E. Francesconi, S. Montemagni, W. Peters, and D. Tiscornia, editors, *Semantic Processing of Legal Texts: Where the Language of Law Meets the Law of Language*, pages 192–215. Springer Berlin Heidelberg, 2010.

- 
- [15] J. Mullenbach, S. Wiegrefe, J. Duke, J. Sun, and J. Eisenstein. 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), pages 1101–1111, 2018.
- [16] S. A. P. Parambath, N. Usunier, and Y. Grandvalet. Optimizing f-measures by cost-sensitive classification. In Advances in Neural Information Processing Systems, volume 27, 2014.
- [17] B. Perozzi, R. Al-Rfou, and S. Skiena. Deepwalk: Online learning of social representations. In Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), pages 701–710, 2014.
- [18] I. Pillai, G. Fumera, and F. Roli. Threshold optimisation for multi-label classifiers. *Pattern Recognition*, 46(7):2055–2065, 2013.
- [19] F. Provost. Machine learning from imbalanced data sets 101. In Proceedings of the AAAI Workshop on Imbalanced Data Sets, pages 1–3, 2000.
- [20] E. Schultheis, M. Wydmuch, R. Babbar, and K. Dembczynski. 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), pages 1547–1557, 2022.
- [21] F. Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1):1–47, 2002.
- [22] S. Shalev-Shwartz, Y. Singer, and N. Srebro. Pegasos: primal estimated sub-gradient solver for SVM. In Proceedings of the Twenty Fourth International Conference on Machine Learning (ICML), 2007.
- [23] A. Sun, E.-P. Lim, and Y. Liu. On strategies for imbalanced text classification using svm: A comparative study. *Decision Support Systems*, 48(1):191–201, 2009.

- 
- [24] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei. Line: Large-scale information network embedding. In Proceedings of the 24th international Conference on World Wide Web (WWW), pages 1067–1077, 2015.
- [25] L. Tang and H. Liu. Relational learning via latent social dimensions. In Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD), pages 817–826, 2009.
- [26] G. Wu and E. Y. Chang. Class-boundary alignment for imbalanced dataset learning. In ICML Workshop on Learning from Imbalanced Data Sets II, pages 49–56, 2003.
- [27] Y. Yang. A study on thresholding strategies for text categorization. In W. B. Croft, D. J. Harper, D. H. Kraft, and J. Zobel, editors, Proceedings of the 24th ACM International Conference on Research and Development in Information Retrieval, pages 137–145, New Orleans, US, 2001. ACM Press, New York, US.
- [28] H.-F. Yu, K. Zhong, J. Zhang, W.-C. Chang, and I. S. Dhillon. PECOS: Prediction for enormous and correlated output spaces. *Journal of Machine Learning Research*, 23(98):1–32, 2022.
- [29] G.-X. Yuan, K.-W. Chang, C.-J. Hsieh, and C.-J. Lin. A comparison of optimization methods and software for large-scale l_1 -regularized linear classification. *Journal of Machine Learning Research*, 11:3183–3234, 2010.
- [30] J. Zhang, W.-C. Chang, H.-F. Yu, and I. S. Dhillon. Fast multi-resolution transformer fine-tuning for extreme multi-label text classification. 34:7267–7280, 2021.
- [31] A. Zubiaga. Enhancing navigation on Wikipedia with social tags. In Proceedings of Wikimania, 2009.



Appendix A

Proofs

A.1 Proof of Theorem 1

Proof. We define $F^*(tp^*, fp^*, fn^*; a, b)$ 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. Similarly, we define $\hat{F}(\hat{tp}, \hat{fp}, \hat{fn}; a, b)$ to be the smoothed F-measure attained by setting the threshold, say \hat{T} , higher than all scores, where \hat{tp} , \hat{fp} and \hat{fn} denotes the counts corresponding to \hat{T} . Note that

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

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 further derive the following relation between tp^* and fp^* :

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

□



A.2 Proof of Corollary 2

Proof. If $a > 2(b + p)$, then we have

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

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

A.3 Proof of Theorem 3

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

$$a < \frac{2(b + p)}{N} \implies T^* \leq z. \quad (\text{A.2})$$

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

$$F(T; a, b) = \frac{a + 2tp(T)}{b + p + tp(T) + fp(T)} \geq \frac{a + 2}{b + p + N} \quad (\text{A.3})$$

because $tp(T) \geq 1$ and $tp(T) + fp(T) \leq N$. For any $T > z$, the corresponding smoothed F-measure satisfies

$$F(T; a, b) = \frac{a + 2tp(T)}{b + p + tp(T) + fp(T)} \leq \frac{a}{b + p} \quad (\text{A.4})$$

since $\text{tp}(T) = 0$ and $\text{fp}(T) \geq 0$. From (A.3) and (A.4), if we have

$$\frac{a+2}{b+p+N} > \frac{a}{b+p}, \quad (\text{A.5})$$

then $T^* \leq z$, our target in (A.2), must be true. Inequality (A.5) is equivalent to

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

□

A.4 Proof of Theorem 4

Proof. Without loss of generality, we assume

$$\phi(x_1), \phi(x_2), \dots, \phi(x_N)$$

are sorted. That is, we have $\phi(x_i) < \phi(x_{i+1})$ for $i = 1, \dots, N-1$. Then the optimization of smoothed F-measure is done by searching through these values of thresholds (as in Algorithm 2.1):

(T_0, T_1, \dots, T_N) where

$$T_i = \begin{cases} \phi(x_1) - \epsilon & i = 0 \\ \frac{1}{2} (\phi(x_i) + \phi(x_{i+1})) & 0 < i < N \\ \phi(x_N) + \epsilon & i = N \end{cases}$$

for some small constant $\epsilon > 0$

We use $F(T_i)$ and $F(T_i; a, b)$ to denote the original F-measure and smoothed F-measure given by the threshold T_i , respectively. Suppose a and b satisfy

$$\frac{a}{b+p} \geq fbr + \frac{a}{b+p+1}, \quad (\text{A.6})$$

we show that the smoothed F-measure reaches optimum at T_N if

$$\forall T_i \ F(T_i) < fbr. \quad (\text{A.7})$$



From the assumptions, we have

$$\begin{aligned}
 F(T_N; a, b) &= \frac{a}{b + p} \\
 &\geq fbr + \frac{a}{b + p + 1} && \text{(by (A.6))} \\
 &> \max_{i=0, \dots, N-1} F(T_i) + \frac{a}{b + p + 1} && \text{(by (A.7))} \\
 &\geq \max_{i=0, \dots, N-1} \frac{2tp(T_i)}{p + tp(T_i) + fp(T_i)} \\
 &\quad + \max_{i=0, \dots, N-1} \frac{a}{b + p + tp(T_i) + fp(T_i)} \\
 &\geq \max_{i=0, \dots, N-1} \frac{2tp(T_i)}{b + p + tp(T_i) + fp(T_i)} \\
 &\quad + \max_{i=0, \dots, N-1} \frac{a}{b + p + tp(T_i) + fp(T_i)} \\
 &\geq \max_{i=0, \dots, N-1} \frac{a + 2tp(T_i)}{b + p + tp(T_i) + fp(T_i)} \\
 &= \max_{i=0, \dots, N-1} F(T_i; a, b).
 \end{aligned}$$

That is, optimizing smoothed F-measure would set the threshold to T_N , which is higher than all decision values. The condition (A.6) is equivalent to

$$a \geq fbr \cdot (b + p)(b + p + 1), \quad (\text{A.8})$$

which can certainly be satisfied by some a and b .

Note that in the extreme case of large a , the algorithm always sets the threshold higher than all decision values, as derived in Corollary 2. Therefore, by picking a large a , the theorem statement is actually trivially true. However, the bound derived in (A.8) is not trivial. For small values of fbr , a can be small enough not to degenerate into that extreme behavior. □

A.5 Proof of Theorem 5

Proof. Let i' be the index satisfying assumption (4.3). 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 $T \geq T'$, we have $\text{fp}(T) = 0$ by the assumption in (4.3). Therefore, for these 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 and tp decreases, the smoothed F-measure also decreases. Therefore, the optimal threshold T^* must be less than $\phi(x_{i'})$. □



Appendix B

Implementation details of micromacro

In this supplementary section, we describe how to evaluate (4.5) efficiently in $O(1)$ time so that the overall complexity of micromacro is $O(L)$. When we are tuning the threshold T_j of the j th label, the optimization problem to be solved is

$$T_j^* = \arg \max_{T_j} F_{1,j}^{\text{macro}} + F_{1,j}^{\text{micro}}.$$

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

$$F_{1,j}^{\text{macro}} = \frac{1}{j} (\underbrace{F_{1,j-1}}_{\text{constant}} + F_j) \quad (\text{B.1})$$

When training problem j , the F-measure $F_{1,j-1}$ of problem 1 to $j - 1$ is an additive constant. Therefore, removing it would still give the same result. This allows macro-F to be evaluated efficiently on the current binary problem, disregarding j . For micro-F, the measure can be rewritten as follows:

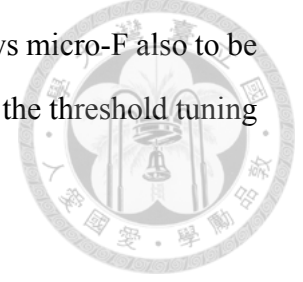
$$\begin{aligned} F_{1,j}^{\text{micro}} &= \frac{2 \text{tp}_{1,j}}{2 \text{tp}_{1,j} + \text{fp}_{1,j} + \text{fn}_{1,j}} \\ &= \frac{2\text{tp}_{1,j-1} + 2\text{tp}_j}{(2 \text{tp}_{1,j-1} + \text{fp}_{1,j-1} + \text{fn}_{1,j-1}) + (2 \text{tp}_j + \text{fp}_j + \text{fn}_j)} \end{aligned} \quad (\text{B.2})$$

We denote the sum of true positives for binary problems k to j as $\text{tp}_{k,j}$ and the number of true positives solely for problem j as tp_j . The same goes for false positives and false

negatives. When optimizing $F_{1,j}^{\text{micro}}$ by tuning the threshold T_j , the accumulated counts $\text{tp}_{1,j-1}$, $\text{fp}_{1,j-1}$ and $\text{fn}_{1,j-1}$ can be pre-calculated and fixed. This allows micro-F also to be evaluated efficiently without looping over the labels. After finishing the threshold tuning of the j th label, the accumulated counts can be updated by

$$\text{tp}_{1,j} = \text{tp}_{1,j-1} + \text{tp}_j^*$$

where tp_j^* is the count generated by the optimal threshold T_j^* . The update for $\text{fp}_{1,j}$ and $\text{fn}_{1,j}$ is similar. After the update, the objective for the $j + 1$ th problem can then be evaluated efficiently.





Appendix C

Details of main experiments

Datasets. For all six datasets, we use the preprocessed version available on the LIBSVM Data¹ 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 technique is available. We adopt the representations learned with DeepWalk [17], Node2vec [5] and LINE [24], respectively for PPI, Flickr, and BlogCatalog. Since no standard testing subset is available for these datasets, so we randomly split them into training (80%) and testing set (20%). The model is then trained on the training set and the result on the testing set is reported.

Parameters. 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.

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

For the search range of fbr in SCutFBR and SCutFBR.n, we follow the settings in [11]. That is, fbr is searched over

$$0.1, 0.2, \dots, 0.8.$$

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



Table C.1: Table of main results for node classification datasets, augmented with the standard variations. The result is calculated from 5 random seeds.

method	PPI		Flickr		BlogCatalog	
	micro	macro	micro	macro	micro	macro
binary	44.58±0.00	16.86±0.00	23.26±0.00	13.21±0.00	29.00±0.00	13.28±0.00
SCut	53.58±0.04	48.35±0.01	37.91±0.13	30.16±0.07	32.50±1.88	26.10±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±0.00	37.43±0.10	29.33±0.09	33.46±0.44	25.18±0.31
cost-sensitive-micro	56.14±0.00	40.26±0.00	39.21±0.00	23.86±0.00	39.97±0.00	23.36±0.00
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±0.10	48.35±0.01	37.77±0.25	30.56±0.11	32.15±1.47	26.31±0.38
smooth-group	53.57±0.06	48.35±0.01	37.51±0.12	30.59±0.14	32.20±1.70	26.18±0.39
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 C.2: Table of main results for text classification datasets, augmented with the standard variations. The result is calculated from 5 random seeds.

method	RCV1-topics		EUR-Lex		Wiki10-31K	
	micro	macro	micro	macro	micro	macro
binary	80.39±0.00	49.57±0.00	52.63±0.00	17.40±0.00	26.22±0.00	2.06±0.00
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	12.30±0.03
cost-sensitive	80.97±0.02	56.00±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±0.00	25.56±0.00	33.16±0.00	6.46±0.00
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
smooth-group	81.27±0.10	62.41±0.37	56.69±0.17	28.59±0.25	30.80±0.45	12.69±0.04
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

For the search grid of C^+ in cost-sensitive methods, we follow the settings in [12], so C^+ is searched over

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

Next, we provide the details of a and b searched in smooth-based methods. Using the implementation technique mentioned at the end of Section 4.1, it allows us to search many pairs of a and b with little impact on training time. For all smoothing-based method that needs to select a and b , 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

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

where $\Delta = \sqrt[29]{N}$ and N is the size of 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 (in smooth-all and smooth-group), we use the smallest p in the group to determine the bound since smoothing is mainly for rare labels. In smooth-group, we assign a label with p positive examples to the group with index

$$\lceil (\log_{10}(1+p)) \rceil,$$

so labels with similar p (in log scale) are grouped together. Except for the mentioned parameters, other parameters (e.g., C for SVM) are not tuned and left as default.

Results with standard deviations. The results are in Table C.1 and Table C.2.



Appendix D

Details of Auxiliary Experiments

In this section, we explain how we tested for the occurrences of Figure 3.2 and Figure 3.1 in Section 5.4.

D.1 Occurences of Figure 3.2

We performed a 3-fold CV for medium to frequent labels ($10 \leq p \leq 1000$). For each validation set, we check the following conditions:

1. There are easy positives (positive samples that are not below any negative samples) in the validation set.
2. The FBR heuristic sets the threshold higher than the easy positives.

D.2 Occurences of Figure 3.1

we perform a 3-fold CV for all rare labels ($p < 10$). 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.