Big Data Research ••• (••••) •••–•••

Contents lists available at ScienceDirect

Big Data Research

www.elsevier.com/locate/bdr

Finding the Best Classification Threshold in Imbalanced Classification ☆

Quan Zou [a,b], Sifa Xie [b], Ziyu Lin [b], Meihong Wu [b], Ying Ju [b]

[a] School of Computer Science and Technology, Tianjin University, Tianjin, China
[b] Department of Computer Science, Xiamen University, Xiamen, China

ARTICLE INFO

Article history:
Received 26 August 2015
Received in revised form 22 October 2015
Accepted 1 December 2015
Available online xxxx

Keywords:
Receiver Operating Characteristic (ROC)
Protein remote homology detection
Imbalance data
F-score

ABSTRACT

Classification with imbalanced class distributions is a major problem in machine learning. Researchers have given considerable attention to the applications in many real-world scenarios. Although several works have utilized the area under the receiver operating characteristic (ROC) curve to select potentially optimal classifiers in imbalanced classifications, limited studies have been devoted to finding the classification threshold for testing or unknown datasets. In general, the classification threshold is simply set to 0.5, which is usually unsuitable for an imbalanced classification. In this study, we analyze the drawbacks of using ROC as the sole measure of imbalance in data classification problems. In addition, a novel framework for finding the best classification threshold is proposed. Experiments with SCOP v.1.53 data reveal that, with the default threshold set to 0.5, our proposed framework demonstrated a 20.63% improvement in terms of F-score compared with that of more commonly used methods. The findings suggest that the proposed framework is both effective and efficient. A web server and software tools are available via http://datamining.xmu.edu.cn/prht/ or http://prht.sinaapp.com/.

1. Background

A dataset is imbalanced if it contains a small amount of samples in one class as compared with the rest of the classes. Without loss of generality, a minority class is regarded as a positive class, whereas a majority class is viewed as a negative class. Imbalanced classification is one of most popular topics in the field of machine learning [1–4]. This issue is represented in many real-world applications, such as bioinformatics [5–11], telecommunications management [12], text classification [13], face recognition [14], and ozone level forecasting [15]. Traditional classifications algorithms perform poorly on imbalanced datasets because the applied evaluation metrics, such as the overall accuracy metric, force classifiers to minimize the error rate, i.e., the percentage of the incorrect prediction of class labels. As a result, classifiers demonstrate good accuracy on the majority class but poor accuracy on the minority class. However, in most imbalanced classification problems, the misclassification error of the minority class is far costlier than that of the majority class. For example, in the medical diagnosis of a certain cancer, misclassifying a cancer patient as healthy is more serious than misclassifying a non-cancer patient as unhealthy, because, in the former, the patient might lose his/her life.

Table 1
Confusion matrix for binary classification.

	Positive prediction	Negative prediction
Positive class	True positive (TP)	False negative (FN)
Negative class	False positive (FP)	True negative (TN)

$\text{TPrate} = \frac{TP}{TP+FN}$, the percentage of positive samples correctly classified.
$\text{TNrate} = \frac{TN}{FP+TN}$, the percentage of negative samples correctly classified.
$\text{FPrate} = \frac{FP}{FP+TN}$, the percentage of negative samples misclassified.
$\text{FNrate} = \frac{FN}{TP+FN}$, the percentage of positive samples misclassified.

As previously mentioned, in imbalanced domains, a specific metric is needed to evaluate the performance of the classifier. The receiver operating characteristic (ROC) graphic [16–19] is commonly used as an evaluation criterion. For a binary classification, we can obtain a confusion matrix, as shown in Table 1, and based on which, four metrics can be calculated.

The ROC graphic depicts the trade-off between benefits (TPrate) and costs (FPrate); in other words, one classifier cannot increase the number of true positives without increasing the false positives. In a ROC curve, the x-axis represents the FPrate and the y-axis represents the TPrate. The points in the curve are obtained by sweeping the classification threshold from the most positive classification value to the most negative. The area under the ROC curve (AUC) [20] is a useful metric for classifying performance because it gives the probability that a randomly selected pair of samples (one

☆ This article belongs to Analytics and Applications.

E-mail address: yju@xmu.edu.cn (Y. Ju).

http://dx.doi.org/10.1016/j.bdr.2015.12.001
2214-5796/

positive and one negative) would have their predicted probabilities correctly ordered.

In imbalanced classification domains, ROCs are considered the "gold standard" of a classifier's ability. However, using only the ROC to select a potentially optimal classifier is not enough. In fact, the ROC curve and the AUC values reflect only the ranking power of positive prediction probability. Furthermore, a high AUC does not insure a high prediction accuracy. For example, in a dataset containing only 1% positive samples, the AUC value can reach more than 0.9 only if all the positive samples rank in the top 10% according to prediction probability. Even if the probabilities of positive samples are predicted to be less than 0.5, as long as the positive probabilities exceed the negative ones, the ROC will exhibit good performance. This phenomenon is typical in imbalanced datasets. Therefore, finding an appropriate prediction probability threshold is as important as a perfect ROC curve for the accurate prediction of testing and unknown data. In most classifiers, the default prediction probability threshold is 0.5. However, this threshold does not work well for imbalanced classification prediction.

Although researchers have attempted to raise the AUC value in previous works, these investigations disregarded the prediction probability thresholds for testing and unknown data. Consequently, classification performance, including recall, precision, and F-scores, remains imperfect even if the AUC value could become rather high. Few tools or Web servers are available for finding the classification threshold. In this paper, we propose a sampling-based threshold auto-tuning method to address this problem. This method can obtain perfect performance on the AUC criteria in addition to very good precision, recall, and F-scores.

2. Methods

2.1. F-score should be another metric aside from the AUC

The AUC is often considered a reliable performance metric for imbalanced binary classification problems [21–24]. However, when the dataset is imbalanced and the AUC has reached a high score, the classification performance may not be as perfect as the AUC value reflects because plenty of "trash" negative samples exist in the imbalanced dataset. "Trash" negative samples raise the AUC value, but a few other negative samples remain mixed with the positive samples, which are difficult to distinguish. These few remaining negative samples diminish performance, including precision and recall, while very slightly influencing the AUC value. In the testing dataset, the values of precision and recall may be less than 0.5, whereas the AUC value can exceed 0.9. AUC50 was proposed to address this problem and to measure the performance of protein remote homology detection [25] (Fig. 1). The AUC50 refers to the AUC up to the first 50 false positive samples. Although the AUC50 can avoid the influence of "trash" true negative samples, 50 is overly arbitrary for various datasets. If less than 50 true negative samples exist in the dataset, then the AUC50 is equal to the AUC. Furthermore, if the training samples are massive, and 50 false positive samples account for only a very small portion of the training set, the AUC50 would be meaningless. Therefore, even though the AUC50 can often better describe classification performance than the AUC, it cannot alleviate the problem inherent to massive data. Thus, we need a different metric altogether along with the AUC to measure classification performance.

We attempt to employ the F-score together with the AUC as a classification measurement for protein remote homology detection. The F-score is a trade-off between precision (P) and recall (R) and is described as follows:

$$P = \frac{\text{TruePositive}}{\text{TruePositive} + \text{FalsePositive}}; \tag{1}$$

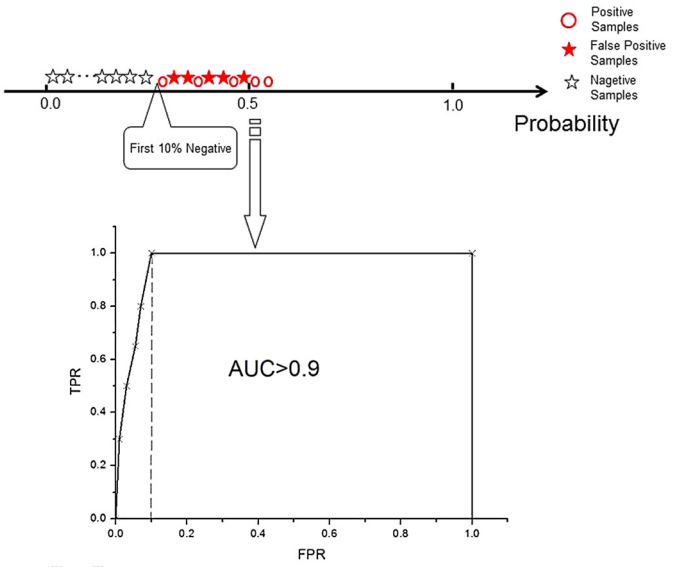

Fig. 1. The ROC of an imbalanced dataset.

$$R = \frac{\text{TruePositive}}{\text{Truepositive} + \text{FalseNegative}}; \tag{2}$$

and

$$F_\beta = \frac{(\beta^2 + 1)PR}{\beta^2 P + R}, \tag{3}$$

where β is a parameter used to adjust the weight between P and R.

2.2. How to set the classification threshold for the testing set

Prediction results are ultimately determined according to prediction probabilities. The threshold is typically set to 0.5. If the prediction probability exceeds 0.5, the sample is predicted to be positive; otherwise, negative. However, 0.5 is not ideal for some cases, particularly for imbalanced datasets.

The probability threshold for classification will not interfere with the AUC value. In other words, the AUC is influenced by the probability ranking result only, and it is not related to the setting of the classification threshold. Therefore, we only need to tune the threshold to obtain the best F-score.

The threshold for the best F-score can be easily obtained if the training set is not massive. We can test all of the probabilities for every positive sample with a brute-force attack. Then, the threshold with the best F-score for the training set can be calculated by using cross-validation [26]. We then determine if the calculated threshold can be used for the test data.

We observed that Liao's protein remote homology detection dataset was not massive enough. The prediction probabilities are distributed differently between the training and testing sets. Moreover, the probability ranges are considerably different, as shown in Fig. 2. We posit that the best threshold position in the training set should be mapped to the corresponding position in the testing set.

We denote the maximum prediction probability in the training set as Maxtrain. In this paper, prediction probability refers to the probability of positive predictions by the classifier. If the prediction probability is less than the threshold, the sample is predicted to be negative. Similarly, we also denote Mintrain, Maxtest, Mintest. Thresholdtrain, and Thresholdtest. Thus, the mapping rule should satisfy the following equation, from which we can compute the Thresholdtest:

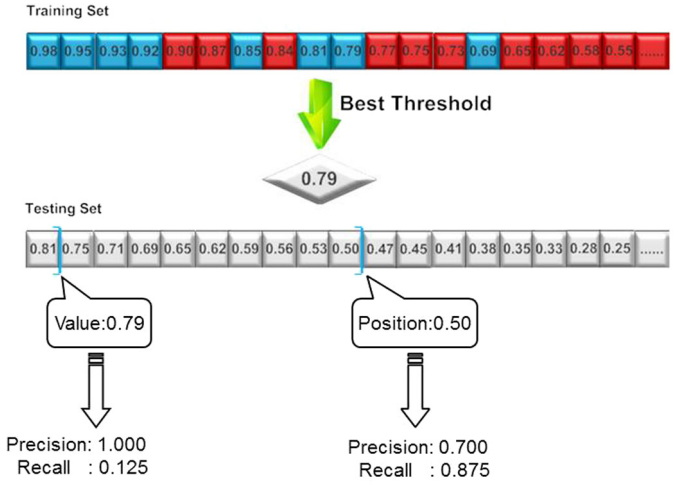

Fig. 2. Two ways to use the best threshold.

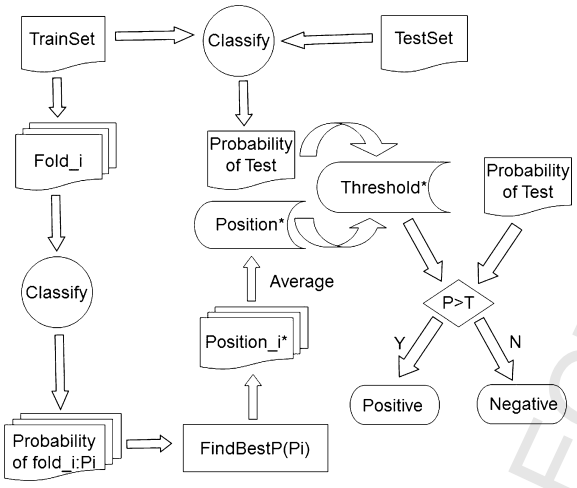

Fig. 3. A flow chart of our proposed method.

$$\text{Thresholdtest} = \left(\frac{\text{Maxtrain} - \text{Thresholdtrain}}{\text{Maxtrain} - \text{Mintrain}}\right) \times (\text{Maxtest} - \text{Mintest}) \tag{4}$$

We can compute the threshold for the training set by using cross-validation and then obtain the threshold for the testing set by using Equation (4). The complete classification model is presented in Fig. 3.

2.3. How to set the classification threshold for a massive training set

A brute-force method can find the best threshold; however, it cannot deal with massive scale data. In this section, we propose a novel algorithm that can quickly compute the best threshold for a massive training set.

We denote x_i as a sample, C_p as the positive samples set, and C_n as the negative samples set. Let $X = (\chi_{\sigma 1}, \chi_{\sigma 2}, \ldots, \chi_{\sigma n})$ be the descending sorted samples set. If we restrict $i < j$, then $P_{\chi \in C_P}(\chi_{\sigma_i}) > P_{\chi \in C_P}(\chi_{\sigma_j})$, and let P_i be the number of positive samples that have a probability that is not less than that of sample χ_{σ_i}. In other words, $P_i = \{|\chi_{\sigma_j}| \mid j \le i \ \& \ \chi_{\sigma_j} \in C_P\}$. n_i is the number of samples that have a probability of not less than that of sample χ_{σ_i}. If χ_{σ_i} is selected to be the threshold sample, then $F_{\beta_i} = \frac{(\beta^2+1)*\frac{p_i}{n_i}*\frac{P_i}{|C_p|}}{\beta^2*\frac{p_i}{n_i}+\frac{p_i}{|C_p|}}$, which yields the following inferences:

Inference 1. The best threshold can be provided by positive samples only.

Proof. Assume that the best threshold is generated by χ_{σ_i} and χ_{σ_i} is a negative sample, then

$$F_{\beta_{Best}} = \frac{(\beta^2+1)*\frac{p_i}{n_i}*\frac{P_i}{|C_p|}}{\beta^2*\frac{p_i}{n_i}+\frac{p_i}{|C_p|}}.$$

Then, select the least positive sample that has a probability that is higher than χ_{σ_i}. If we select the least positive sample as χ_{σ_j}, then $p_i = p_j$, $n_i > n_j$, i.e.,

$$\begin{aligned} F_{\beta_{Best}} &= \frac{(\beta^2+1)*\frac{p_i}{n_i}*\frac{P_i}{|C_p|}}{\beta^2*\frac{p_i}{n_i}+\frac{p_i}{|C_p|}} = \frac{(\beta^2+1)*p_i*\frac{p_i}{|C_p|}}{\beta^2*p_i+\frac{p_i*n_i}{|C_p|}} \\ &< \frac{(\beta^2+1)*p_j*\frac{p_j}{|C_p|}}{\beta^2*p_j+\frac{p_j*n_j}{|C_p|}} = \frac{(\beta^2+1)*\frac{p_j}{n_j}*\frac{p_j}{|C_p|}}{\beta^2*\frac{p_j}{n_j}+\frac{p_j}{|C_p|}} = F_{\beta_i}. \end{aligned} \tag{5}$$

The above equation contradicts χ_{σ_i} as the best threshold sample; therefore, the best threshold can only be generated by positive samples. □

Inference 2. If χ_{σ_i} and $\chi_{\sigma_{i+1}}$ are both positive samples, then $F_{\beta_i} < F_{\beta_{i+1}}$.

Proof. $p_{i+1} = p_i + 1$, $n_{i+1} = n_i + 1$ because χ_{σ_i} and $\chi_{\sigma_{i+1}}$ are both positive samples, i.e.,

$$\begin{aligned} F_{\beta_{i+1}} &= \frac{(\beta^2+1)*\frac{p_{i+1}}{n_{i+1}}*\frac{p_{i+1}}{|C_p|}}{\beta^2*\frac{p_{i+1}}{n_{i+1}}+\frac{p_{i+1}}{|C_p|}} = \frac{(\beta^2+1)*\frac{p_{i+1}}{|C_p|}}{\beta^2+\frac{n_{i+1}}{|C_p|}} \\ &= \frac{(\beta^2+1)*\frac{p_i}{|C_p|}*\frac{p_{i+1}}{p_i}}{\beta^2+\frac{n_i}{|C_p|}*\frac{n_{i+1}}{n_i}} \\ &= \frac{(\beta^2+1)*\frac{p_i}{|C_p|}}{\frac{n_i}{n_{i+1}}*\beta^2+\frac{n_i}{|C_p|}}*\frac{\frac{p_{i+1}}{p_i}}{\frac{n_{i+1}}{n_i}} \\ &> \frac{(\beta^2+1)*\frac{p_i}{|C_p|}}{\frac{n_i}{n_{i+1}}*\beta^2+\frac{n_i}{|C_p|}}*\frac{1+\frac{1}{p_i}}{1+\frac{1}{n_i}} \ge \frac{(\beta^2+1)*\frac{p_i}{|C_p|}}{\beta^2+\frac{n_i}{|C_p|}} \\ &= \frac{(\beta^2+1)*\frac{p_i}{n_i}*\frac{p_i}{|C_p|}}{\beta^2*\frac{p_i}{n_i}+\frac{p_i}{|C_p|}} = F_{\beta_i}. \end{aligned} \tag{6}$$

Denote $\chi_{\sigma_{p_j}}$ as the positive sample of j in X, and $F_{\beta_{p_j}}$ as the F value of $\chi_{\sigma_{p_j}}$. Then, from Inference 1, we obtain $F_{\beta_{\text{Best}}} \in \{F_{\beta_{p_j}} \mid j = 1, 2, \ldots, |C_p|\}$ and can derive Inference 3 as follows. □

Inference 3. If $\chi_{\sigma_i} = \chi_{\sigma_{p_j}}$ is the positive samples of j, and the next κ samples are all negative samples, then the maximum $F_{\beta_{p_t}}(t > j)$ is $\frac{(\beta^2+1)*\frac{|C_p|}{n_i+k+|C_p|-p_i}}{\beta^2*\frac{|C_p|}{n_i+k+|C_p|-p_i}+1}$.

Proof. $F_{\beta_{p_{j+1}}}$ receives its maximum value when the sample of $(i+k+1)$ is positive. Suppose y negative samples exist between $\chi_{\sigma_{i+k}}$ and $\chi_{\sigma_{p_{j+1}}}$, then

$$F_{\beta_{p_{j+1}}} = \frac{(\beta^2+1)*\frac{p_i+1}{n_i+k+y+1}*\frac{p_i+1}{|C_p|}}{\beta^2*\frac{p_i+1}{n_i+k+y+1}+\frac{p_i+1}{|C_p|}} = \frac{(\beta^2+1)*\frac{p_i+1}{|C_p|}}{\beta^2+\frac{n_i+k+y+1}{|C_p|}}, \tag{7}$$

when $y = 0$, $F_{\beta_{p_{j+1}}}$ obtains the maximum value: $\frac{(\beta^2+1)*\frac{p_i+1}{|C_p|}}{\beta^2+\frac{n_i+k+1}{|C_p|}}$.

Please cite this article in press as: Q. Zou et al., Finding the Best Classification Threshold in Imbalanced Classification, Big Data Res. (2016), http://dx.doi.org/10.1016/j.bdr.2015.12.001

In this manner, the maximum of $F_{\beta_{p_{j+2}}}$ is $\frac{(\beta^2+1)*\frac{p_i+2}{|C_p|}}{\beta^2+\frac{n_i+k+2}{|C_p|}}$ and so on.

In the last equation, the maximum of $F_{\beta_{p_{|C_p|}}}$ is $\frac{(\beta^2+1)*\frac{|C_p|}{n_i+k+|C_p|-p_i}}{\beta^2*\frac{|C_p|}{n_i+k+|C_p|-p_i}+1}$.

Moreover, with reference to Inference 2, $F_{\beta_{p_{j+1}}} < F_{\beta_{p_{j+2}}} < \cdots < F_{\beta_{p_{|C_p|}}}$; thus, Inference 3 is proven. □

Inference 4. If F_{β_i} is the maximum value so far, and next k samples are all negative samples, then χ_{σ_i} is the best threshold sample, where $k = \frac{|C_p|-p_i}{p_i}(\beta^2 * |C_p| + n_i - p_i)$.

Proof. Based on Inference 3, the maximum F_β after F_{β_i} is $F_{\beta_{p_{|C_p|}}} = \frac{(\beta^2+1)*\frac{|C_p|}{n_i+k+|C_p|-p_i}}{\beta^2*\frac{|C_p|}{n_i+k+|C_p|-p_i}+1}$.

Let $\chi = k + |C_p| - p_i$, and solve the inequality:

$$F_{\beta_i} = \frac{(\beta^2+1)*\frac{p_i}{n_i}*\frac{P_i}{|C_p|}}{\beta^2*\frac{p_i}{n_i}+\frac{p_i}{|C_p|}} > F_{\beta_{p_{|C_p|}}}$$

$$= \frac{(\beta^2+1)*\frac{|C_p|}{n_i+k+|C_p|-p_i}}{\beta^2*\frac{|C_p|}{n_i+k+|C_p|-p_i}+1} = \frac{(\beta^2+1)*\frac{|C_p|}{n_i+\chi}}{\beta^2*\frac{|C_p|}{n_i+\chi}+1}. \quad (8)$$

We can obtain $\chi = \frac{|C_p|-p_i}{p_i}(|C_p| * \beta^2 + n_i)$; therefore, by combining this equation with $\chi = k + |C_p| - p_i$, we obtain

$$k = \frac{|C_p| - p_i}{p_i}\left(\beta^2 * |C_p| + n_i - p_i\right). \quad (9)$$

Given a set of samples, and by applying the above inferences, we propose an efficient method for finding the best threshold of the set. We present a pseudocode for that method in Table 2. The time complexity of the proposed method is $\theta(n)$. □

3. Results and discussion

3.1. Dataset

A commonly used benchmark dataset [27] was used to evaluate the performance of the proposed method. This benchmark dataset has been used to evaluate the performance of various homology detection methods [28–34]. The dataset contains 4352 proteins derived from the SCOP database version 1.53. These proteins were extracted from the Astral database [35], and the sequence similarity of any pair is less than an E-value of 10^{-25}. The 4352 distinct protein sequences are classified into 54 families. In each family, positive testing samples are derived from proteins within that family. Proteins outside the same family but within the same superfamily are considered positive training samples. Protein sequences outside the same superfamily are selected as negative samples and are separated into training and testing sets. Basic information about the 54 families is provided in Table 3.

Table 3 shows that all of the families are imbalanced datasets. This imbalance is reflected by the fact that some families contain a negative-to-positive ratio of as high as 241:1, whereas some families only have 10 positive samples in their training set.

Previous studies showed that the features extracted from protein, DNA, or RNA sequences are efficient for constructing a computational predictor [36,37]. Therefore, in this study we used a 188-feature model [38,39] to extract feature vectors from protein sequences in our experiment. A random forest method was employed as the classification algorithm. The ROC of each family is listed in Table 4.

Table 2
Pseudocodes for finding the best threshold.

Find best threshold
Input: a sorted set of samples $X = (\chi_{\sigma 1}, \chi_{\sigma 2}, \ldots, \chi_{\sigma n})$, β value, the number of positive samples $\lvert C_p \rvert$
Output: Best threshold T^*
Procedure:
1. begin
2. Init $p_i \leftarrow 1, n_i \leftarrow 1, k \leftarrow 0, i \leftarrow 1$;
3. while χ_{σ_i} is negative do
4. $i++$;
5. n_i++;
6. end while
7. $T^* = P(\chi_{\sigma_i})$;
8. $F^* = \frac{(\beta^2+1)*\frac{p_i}{n_i}*\frac{P_i}{\lvert C_p \rvert}}{\beta^2*\frac{p_i}{n_i}+\frac{p_i}{\lvert C_p \rvert}}$
9. for $i \leftarrow i+1$ to n do
10. if χ_{σ_i} is negative
11. $K++$;
12. else
13. if $k >= \frac{\lvert C_p \rvert - p_i}{p_i}(\beta^2 * \lvert C_p \rvert + n_i - p_i)$
14. break;
15. else
16. $n_i \leftarrow n_i + k$;
17. $k \leftarrow 0$;
18. p_i++;
19. $F_{Temp} = \frac{(\beta^2+1)*\frac{p_i}{n_i}*\frac{P_i}{\lvert C_p \rvert}}{\beta^2*\frac{p_i}{n_i}+\frac{p_i}{\lvert C_p \rvert}}$
20. if $F_{Temp} > F^*$
21. $F^* = F_{Temp}$;
22. $T^* = P(\chi_{\sigma_i})$;
23. end if
24. end if
25. end if
26. end for
27. end

4. Experimental methodology

F_1 was used as a measure metric to compare the performance of different thresholds. F_1 is defined as follows:

$$F_1 = \frac{2 * P * R}{P + R} \quad (10)$$

where P and R are the precision and recall of positive samples, respectively.

Initially, a uniform threshold was directly set to divide the testing set. The performance of these different thresholds is shown in Table 5.

As shown in Table 4, the ROC of most families is considerably high; however, when the threshold is set to 0.5, which is the default threshold in most common classifiers, the performance is very poor in terms of precision and recall. In fact, only one family that has samples with probabilities exceeded 0.5 in our experiment. Moreover, the different thresholds yielded different F1 values, suggesting that an appropriate threshold should be set for the testing set.

To improve performance further, we propose a novel method for finding the best threshold specifically for each family. As previously mentioned, the best threshold for the training set with the testing set can be used in two ways: to use the value directly and to use positional information. The performance of these two methods is listed in Table 6 ($\beta = 1$).

When we used the value to divide the testing set, the performance was poor. Although this approach is better than using the commonly employed method of always setting the threshold to 0.5, it is worse compared to when the threshold is uniformly set to 0.1. However, when we used positional information, the performance improved. The precision and recall can reach 0.2051 and 0.3303, respectively. Thus, this method outperforms methods that use a uniform threshold.

Table 3
Basic information regarding the dataset.

Index	ID	Train	Test	Index	ID	Train	Test
1	7.3.5.2	12/2330	9/1746	28	7.3.10.1	11/423	95/3653
2	2.56.1.2	11/2509	8/1824	29	3.32.1.11	46/3880	5/421
3	3.1.8.1	19/3002	8/1263	30	3.32.1.13	43/3627	8/674
4	3.1.8.3	17/2686	10/1579	31	7.3.6.1	33/3203	9/873
5	1.27.1.1	12/2890	6/1444	32	7.3.6.2	16/1553	26/2523
6	1.27.1.2	10/2408	8/1926	33	7.3.6.4	37/3591	5/485
7	3.42.1.1	29/3208	10/1105	34	2.38.4.1	30/3682	5/613
8	1.45.1.2	33/3650	6/663	35	2.1.1.1	90/3102	31/1068
9	1.4.1.1	26/2256	23/1994	36	2.1.1.2	99/3412	22/758
10	2.9.1.2	17/2370	14/1951	37	3.32.1.1	42/3542	9/759
11	1.4.1.2	41/3557	8/693	38	2.38.4.3	24/2946	11/1349
12	2.9.1.3	26/3625	5/696	39	2.1.1.3	113/3895	8/275
13	1.4.1.3	40/3470	9/780	40	2.1.1.4	88/3033	33/1137
14	2.44.1.2	11/307	140/3894	41	2.38.4.5	26/3191	9/1104
15	2.9.1.4	21/2928	10/1393	42	2.1.1.5	94/3240	27/930
16	3.42.1.5	26/2876	13/1437	43	7.39.1.2	20/3204	7/1121
17	3.2.1.2	37/3002	16/1297	44	2.52.1.2	12/3060	5/1275
18	3.42.1.8	34/3761	5/552	45	7.39.1.3	13/2083	14/2242
19	3.2.1.3	44/3569	9/730	46	1.36.1.2	29/3477	7/839
20	3.2.1.4	46/3732	7/567	47	3.32.1.8	40/3374	11/927
21	3.2.1.5	46/3732	7/567	48	1.36.1.5	10/1199	26/3117
22	3.2.1.6	48/3894	5/405	49	7.41.5.1	10/2241	9/2016
23	2.28.1.1	18/1246	44/3044	50	7.41.5.2	10/2241	9/2016
24	3.3.1.2	22/3280	7/1043	51	1.41.1.2	36/3692	6/615
25	3.2.1.7	48/3894	5/405	52	2.5.1.1	13/2345	11/1983
26	2.28.1.3	56/3875	6/415	53	2.5.1.3	14/2525	10/1803
27	3.3.1.5	13/1938	16/2385	54	1.41.1.5	17/1744	25/2563

Table 4
The ROC of each family.

Index	ROC	Index	ROC	Index	ROC	Index	ROC
1	0.973	15	0.982	29	0.717	43	0.823
2	0.809	16	0.690	30	0.883	44	0.766
3	0.983	17	0.871	31	0.972	45	0.848
4	0.963	18	0.768	32	0.971	46	0.820
5	0.785	19	0.860	33	0.998	47	0.925
6	0.964	20	0.948	34	0.587	48	0.901
7	0.791	21	0.982	35	0.936	49	0.833
8	0.894	22	0.986	36	0.983	50	0.981
9	0.949	23	0.666	37	0.933	51	0.913
10	0.950	24	0.667	38	0.569	52	0.847
11	0.961	25	0.895	39	0.908	53	0.819
12	0.986	26	0.644	40	0.952	54	0.943
13	0.998	27	0.683	41	0.757	average	0.861
14	0.517	28	0.981	42	0.790		

Table 5
The performance of different uniform thresholds.

Threshold	MeanPrecision	MeanRecall	MeanF1
0.5	0.01852	0.00013	0.00026
0.3	0.03063	0.00251	0.00463
0.2	**0.17411**	0.04963	0.07171
0.1	0.16850	**0.22259**	**0.18001**

In finding the best threshold for the training set, F_β is used to measure how "well" a threshold is, and β is used to adjust the trade-off between precision and recall. We used different β values and listed the resulting performances in Table 7 (three-fold).

As shown in Table 6, when β increases, the mean recall also increases, whereas the mean precision decreases. Furthermore, the method achieves the best performance when β is set to 0.8, yielding a mean precision of 0.2284 and a mean recall of 0.3091.

Nevertheless, these values for mean precision and mean recall do not seem sufficiently high. Therefore, we designed an experiment that can estimate the best performance possible and explore the upper bound of performance. We directly applied the algorithm to the testing set to discover the true optimal threshold for the testing set. Mean precision and mean recall reached 0.2841 and 0.4944, respectively, which yields a mean F_1 of 0.2923. Compared with the best performing method, the proposed method worked well on mean precision and mean F_1, even though the mean recall is 18.53% lower. For a more detailed comparison, a family-by-family comparison of mean F_1s between performance-bound methods and the proposed method is plotted in Fig. 4. Every point in the graph represents one of the 54 SCOP families. The single point above the diagonal indicates that performance of the proposed method is worse than the best performing method for that particular family.

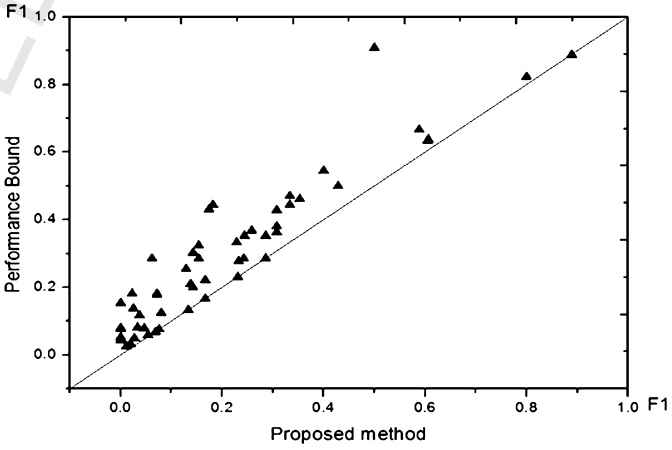

Fig. 4. Family by family comparison of a performance-bound method versus our proposed method.

However, most of the points are located near the diagonal in Fig. 4, thereby indicating that the proposed method can achieve comparable performance as performance-bound methods. In particular, the seven points that lie directly on the diagonal reflect that the proposed method obtained the best performance for those seven families.

The ROC and F_1 of each family are plotted in Fig. 5 to estimate the relation between the ROC and F_1 values. As shown in Fig. 5, the ROC and F1 curves have nearly the same fluctuation

Table 6
The performance of value and position.

Fold	Value			Position		
	MeanPrecision	MeanRecall	MeanF1	MeanPrecision	MeanRecall	MeanF1
2	0.1183	0.1556	0.1053	0.1650	0.2799	0.1593
3	0.1394	0.1272	0.1098	**0.2051**	0.3303	**0.2044**
5	0.1388	0.1473	0.1231	0.1600	0.3430	0.1871
10	0.1324	0.1461	0.1214	0.1557	**0.3882**	0.1896

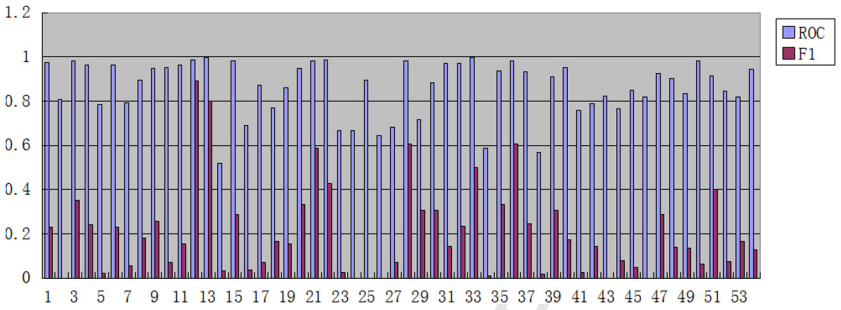

Fig. 5. The distribution of ROC and F1.

Table 7
The performance of different βs.

β	MeanPrecision	MeanRecall	MeanF1
0.5	**0.2452**	0.2633	0.1884
0.8	0.2284	0.3091	**0.2066**
1.0	0.2051	0.3303	0.2044
1.2	0.1948	0.3364	0.2036
1.5	0.1725	**0.3729**	0.1996

trends. The family that has a high ROC would more likely receive a high F1, and the families that have ROC higher than 0.9 all have F1 higher than 0.06, implying that our proposed method is more effective for the testing set that has a high ROC.

5. Conclusions

The disadvantage of using AUC for protein remote homology detection was explored in this study. A novel method was proposed for finding the proper prediction probability threshold of a testing set. Experimental evaluation was performed by using an established benchmark, and the results showed that the proposed method can effectively improve prediction performance over more commonly employed methods. In the future, we intend to explore the efficiency of using a function to classify a testing set, as compared with using a single threshold. We expect that a linear function will achieve better performance. Other approaches should also be employed for finding the proper prediction probability threshold, e.g., neural-like computing models [40–43], Hadoop based methods [44,45], which have widely been used in pattern recognition.

6. Competing interests

The authors declare that they have no competing interests.

7. Authors' contributions

QZ performed the study demonstrating the disadvantage of using ROC as a measurement of protein remote homology detection, and initially drafted the manuscript. SFX established the framework for finding the best testing set threshold, and participated in the design of the study. YJ participated in the design of the study, and performed the statistical analyses. ZYL and MHW conceived of the study, participated in its design and coordination, and helped to draft the manuscript. All authors read and approved the final manuscript.

Acknowledgements

The work was supported by the Natural Science Foundation of China (No. 61370010, No. 61303004), the Natural Science Foundation of Fujian Province of China (No. 2014J01253), and the Major Program of the National Social Science Foundation of China (No. 13&ZD148).

References

[1] Q. Yang, X. Wu, 10 challenging problems in data mining research, Int. J. Inf. Technol. Decis. Mak. 5 (2006) 597–604.
[2] V. López, A. Fernández, S. García, et al., An insight into classification with imbalanced data: empirical results and current trends on using data intrinsic characteristics, Inf. Sci. 250 (2013) 113–141.
[3] Y. Tang, Y.Q. Zhang, N.V. Chawla, et al., SVMs modeling for highly imbalanced classification, IEEE Trans. Syst. Man Cybern., Part B, Cybern. 39 (2009) 281–288.
[4] V. Ganganwar, An overview of classification algorithms for imbalanced datasets, Int. J. Emerging Technol. Adv. Eng. 2 (2012) 42–47.
[5] Li Song, Dapeng Li, Xiangxiang Zeng, Yunfeng Wu, Li Guo, Quan Zou, nDNA-prot: identification of DNA-binding proteins based on unbalanced classification, BMC Bioinform. 15 (2014) 298.
[6] C. Lin, Y. Zou, Ji. Qin, et al., Hierarchical classification of protein folds using a novel ensemble classifier, PLoS ONE 8 (2013) e56499.
[7] Xiangxiang Zeng, Xuan Zhang, Quan Zou, Integrative approaches for predicting microRNA function and prioritizing disease-related microRNA using biological interaction networks, Brief. Bioinform. (2015), http://dx.doi.org/10.1093/bib/bbv033.
[8] Quan Zou, Jinjin Li, Li Song, Xiangxiang Zeng, Guohua Wang, Similarity computation strategies in the microRNA-disease network: a survey, Brief. Funct. Genomics (2015), http://dx.doi.org/10.1093/bfgp/elv024.
[9] B. Liu, J. Xu, X. Lan, R. Xu, J. Zhou, X. Wang, K.-C. Chou, iDNA-Prot|dis: identifying DNA-binding proteins by incorporating amino acid distance-pairs and reduced alphabet profile into the general pseudo amino acid composition, PLoS ONE 9 (9) (2014) e106691.
[10] B. Liu, L. Fang, F. Liu, X. Wang, J. Chen, K.-C. Chou, Identification of real microRNA precursors with a pseudo structure status composition approach, PLoS ONE 10 (3) (2015) e0121501.
[11] Xiangxiang Zeng, Sisi Yuan, Xianxian Huang, Quan Zou, Identification of cytokine via an improved genetic algorithm, Front. Comput. Sci. 9 (4) (2015) 643–651.
[12] K.J. Ezawa, M. Singh, S.W. Norton, Learning goal oriented Bayesian networks for telecommunications risk management, in: Proceedings of the International Conference on Machine Learning, Morgan Kaufmann, Bari, Italy, 1996, pp. 139–147.

[13] D. Lewis, J. Catlett, Heterogeneous uncertainity sampling for supervised learning, in: Proceedings of the Eleventh International Conference of Machine Learning, Morgan Kaufmann, San Francisco, CA, 1994, pp. 148–156.
[14] N. Kwak, Feature extraction for classification problems and its application to face recognition, Pattern Recognit. 41 (2008) 1718–1734.
[15] C.H. Tsai, L.C. Chang, H.C. Chiang, Forecasting of ozone episode days by cost-sensitive neural network methods, Sci. Total Environ. 407 (2009) 2124–2135.
[16] A.P. Bradley, The use of the area under the ROC curve in the evaluation of machine learning algorithms, Pattern Recognit. 30 (1997) 1145–1159.
[17] S. Yang, F. Zheng, X. Luo, S. Cai, Y. Wu, K. Liu, M. Wu, J. Chen, S. Krishnan, Effective dysphonia detection using feature dimension reduction and kernel density estimation for patients with Parkinson's disease, PLoS ONE 9 (2014) e88825.
[18] S. Yang, S. Cai, F. Zheng, Y. Wu, K. Liu, M. Wu, Q. Zou, J. Chen, Representation of fluctuation features in pathological knee joint vibroarthrographic signals using kernel density modeling method, Med. Eng. Phys. 36 (2014) 1305–1311.
[19] R.M. Rangayyan, F. Oloumi, Y. Wu, S. Cai, Fractal analysis of knee-joint vibroarthrographic signals in power spectral analysis, Biomed. Signal Process. Control 8 (2013) 23–29.
[20] J. Huang, C.X. Ling, Using AUC and accuracy in evaluating learning algorithms, IEEE Trans. Knowl. Data Eng. 17 (2005) 299–310.
[21] X.Y. Liu, J.X. Wu, Z.H. Zhou, Exploratory undersampling for class-imbalance learning, IEEE Trans. Syst. Man Cybern., Part B, Cybern. 39 (2009) 539–550.
[22] L. Cheng, Z.G. Hou, Y. Lin, Recurrent neural network for non-smooth convex optimization problems with application to the identification of genetic regulatory networks, IEEE Trans. Neural Netw. 22 (2011) 714–726.
[23] L.Y. Wei, M.H. Liao, R.R. Ji, et al., Improved and promising identification of human MicroRNAs by incorporating a high-quality negative set, IEEE/ACM Trans. Comput. Biol. Bioinform. 11 (2014) 192–201.
[24] B. Liu, L. Fang, C. Jie, F. Liu, X. Wang, MiRNA-dis: microRNA precursor identification based on distance structure status pairs, Mol. BioSyst. 11 (2015) 1194–1204.
[25] B.H. Asa, B. Douglas, Remote homology detection: a motif based approach, Bioinformatics 19 (2003) 26–33.
[26] B. Liu, L. Fang, S. Wang, X. Wang, H. Li, K.-C. Chou, Identification of microRNA precursor with the degenerate K-tuple or Kmer strategy, J. Theor. Biol. 385 (2015) 153–159.
[27] L. Liao, S.N. William, Combining pairwise sequence similarity and support vector machines for detecting remote protein evolutionary and structural relationships, J. Comput. Biol. 10 (2003) 857–868.
[28] Q.W. Dong, X.L. Wang, L. Lin, Application of latent semantic analysis to protein remote homology detection, Bioinformatics 22 (2006) 285–290.
[29] H. Saigo, J.P. Vert, N. Ueda, T. Akutsu, Protein homology detection using string alignment kernels, Bioinformatics 20 (2004) 1682–1689.
[30] T. Lingner, P. Meinicke, Word correlation matrices for protein sequence analysis and remote homology detection, BMC Bioinform. 9 (2008) 259.
[31] B. Liu, D. Zhang, R. Xu, J. Xu, X. Wang, Q. Chen, Q. Dong, K.-C. Chou, Combining evolutionary information extracted from frequency profiles with sequence-based kernels for protein remote homology detection, Bioinformatics 30 (4) (2014) 472–479.
[32] B. Liu, J. Xu, Q. Zou, R. Xu, X. Wang, Q. Chen, Using distances between top-n-gram and residue pairs for protein remote homology detection, BMC Bioinform. 15 (Suppl 2) (2014) S3.
[33] B. Liu, J. Chen, X. Wang, Protein remote homology detection by combining Chou's distance-pair pseudo amino acid composition and principal component analysis, Mol. Gen. Genet. 290 (5) (2015) 1919–1931.
[34] B. Liu, X. Wang, Q. Zou, Q. Dong, Q. Chen, Protein remote homology detection by combining Chou's pseudo amino acid composition and profile-based protein representation, Molecular Inf. 32 (2013) 775–782.
[35] T. Lingner, P. Meinicke, Word correlation matrices for protein sequence analysis and remote homology detection, BMC Bioinform. 9 (2008) 259.
[36] B. Liu, F. Liu, L. Fang, X. Wang, K.-C. Chou, repDNA: a Python package to generate various modes of feature vectors for DNA sequences by incorporating user-defined physicochemical properties and sequence-order effects, Bioinformatics 31 (8) (2015) 1307–1309.
[37] B. Liu, F. Liu, X. Wang, J. Chen, L. Fang, K.-C. Chou, Pse-in-one: a web server for generating various modes of pseudo components of DNA, RNA, and protein sequences, Nucleic Acids Res. (2015) W65–W71.
[38] C.Z. Cai, L.Y. Han, Z.L. Ji, et al., SVM-Prot: web-based support vector machine software for functional classification of a protein from its primary sequence, Nucleic Acids Res. 31 (2003) 3692–3697.
[39] Q. Zou, X.B. Li, Y. Jiang, et al., BinMemPredict: a Web server and software for predicting membrane protein types, Current Proteomics 10 (2013) 2–9.
[40] Xiangrong Liu, Zimin Li, Juan Liu, Logan Liu, Xiangxiang Zeng, Implementation of arithmetic operations with time-free spiking neural P systems, IEEE Trans. NanoBiosci. 14 (6) (2015) 617–624.
[41] Tao Song, Xiangrong Liu, Xiangxiang Zeng, Asynchronous spiking neural P systems with anti-spikes, Neural Proces. Lett. (2015), http://dx.doi.org/10.1007/s11063-014-9378-1.
[42] Tao Song, Quan Zou, Xiangrong Liu, Xiangxiang Zeng, Asynchronous spiking neural P systems with rules on synapses, Neurocomputing 152 (2015) 1439–1445.
[43] Xiangxiang Zeng, Xingyi Zhang, Tao Song, Linqiang Pan, Spiking neural P systems with thresholds, Neural Comput. 26 (7) (2014) 1340–1361.
[44] Quan Zou, Xubin Li, Wenrui Jiang, Ziyu Lin, Guilin Li, Ke Chen, Survey of MapReduce frame operation in bioinformatics, Brief. Bioinform. 15 (4) (2014) 637–647.
[45] Quan Zou, Qinghua Hu, Maozu Guo, Guohua Wang, HAlign: fast multiple similar DNA/RNA sequence alignment based on the centre star strategy, Bioinformatics 31 (15) (2015) 2475–2481.

Please cite this article in press as: Q. Zou et al., Finding the Best Classification Threshold in Imbalanced Classification, Big Data Res. (2016), http://dx.doi.org/10.1016/j.bdr.2015.12.001