Cost-Sensitive Learning Methods for Imbalanced Data

Nguyen Thai-Nghe, Zeno Gantner, and Lars Schmidt-Thieme, *Member, IEEE*

***Abstract*— Class imbalance is one of the challenging problems for machine learning algorithms. When learning from highly imbalanced data, most classifiers are overwhelmed by the majority class examples, so the false negative rate is always high. Although researchers have introduced many methods to deal with this problem, including resampling techniques and cost-sensitive learning (CSL), most of them focus on either of these techniques. This study presents two empirical methods that deal with class imbalance using both resampling and CSL. The first method combines and compares several sampling techniques with CSL using support vector machines (SVM). The second method proposes using CSL by optimizing the cost ratio (cost matrix) locally. Our experimental results on 18 imbalanced datasets from the UCI repository show that the first method can reduce the misclassification costs, and the second method can improve the classifier performance.**

I. Introduction

In binary classification problems, class imbalance can be described as the majority class outnumbering of the minority one by a large factor. This phenomenon appears in many machine learning and data mining applications, such as credit card fraud detection, intrusion detection, oil-spill detection, disease diagnosis, and many other areas. Most classifiers in supervised machine learning are designed to maximize the accuracy of their models. Thus, when learning from imbalanced data, they are usually overwhelmed by the majority class examples. This is the main problem that degrades the performance of such classifiers ([1], [2]). It is also considered as one of ten challenging problems in data mining research [3].

Researchers have introduced many techniques to deal with class imbalance, as summarized in [1] and [2]. Most of them focus on the manipulation at the data level (resampling methods) such as in [4], [5], [6], [7], [8], [9], [10] and the classifier level (changing the classifier internally) such as in [11], [12], [13], [14], [15], [16], [17], [18].

A related problem is cost-sensitive learning (CSL). Many past publications have applied CSL to decision trees ([19], [20], [21], [22]) or Naive Bayes ([23], [24]). In addition, to understand how class imbalance affects CSL, some authors have analyzed the behavior of the classifier (e.g. C4.5) when applying CSL ([21], [25]). Previous works have also combined manipulations at the data-level with classifier-level modifications ([26], [27], [28]).

Although many papers about the class imbalance problem have been written, most of them focus on *either resampling techniques or CSL*. Our contributions consist of two methods which utilize both resampling techniques and CSL.

- The first method combines and compares several sampling techniques with CSL using an SVM as the base classifier. Concretely, in the first step of this combination, we re-balanced the datasets by using some resampling techniques such as TLINK, RUS, ROS, and SMOTE (we will explain these methods in the next section); in the next step, we trained SVM models on these re-balanced datasets. The outputs produced by SVM were fitted by a sigmoid function based on the method by Platt [29] to get the posterior probabilities. Finally, a Bayes risk (conditional risk) criterion was used to get the final model with minimum expected costs.
- The second method for CSL, instead of assuming that we know the cost ratio (or cost matrix) before learning as in the first method and other previous works ([30], [21], [25]) or setting the cost ratio by inverting of prior class distributions ([31], [7], [32]), we treat this number as a hyperparameter, optimize it locally and then train the final models.

Our experiments on 18 imbalanced datasets from UCI show that these methods are useful. The first method helps reducing misclassification costs and the second method helps improving classifier performance (e.g. the GMean metric).

The rest of the paper is organized as follows. Section II introduces some related work; in session III, we summarize some common techniques that are usually used to tackle the class imbalance problem; section IV describes the proposed methods; section V presents the datasets; section VI shows the experimental results; and finally, section VII is the conclusion.

II. Related Work

Many sampling techniques have been introduced including heuristic or non-heuristic oversampling ([4], [5]), undersampling ([6], [7]), and data cleaning rules such as removing "noise" and "borderline" examples ([8], [9], [10]). These works focus on data-level techniques.

Other researchers concentrate on changing the classifier internally, for example SVM, to deal with class imbalance such as [11], [12], and [13]; [14] uses ensemble learning to deal with class imbalance while [15] combines undersampling with ensemble methods; [16] focuses on incorporating different re-balance heuristics to SVM to tackle the problem

Nguyen Thai-Nghe is with the Information Systems and Machine Learning Lab, University of Hildesheim, Samelsonplatz 1, 31141 Hildesheim, Germany (phone: +49 5121 883 765; email: nguyen@ismll.uni-hildesheim.de).

Zeno Gantner is with the Information Systems and Machine Learning Lab, University of Hildesheim, Samelsonplatz 1, 31141 Hildesheim, Germany (phone: +49 5121 883 856; email: gantner@ismll.uni-hildesheim.de).

Lars Schmidt-Thieme is with the Information Systems and Machine Learning Lab, University of Hildesheim, Samelsonplatz 1, 31141 Hildesheim, Germany (phone: +49 5121 883 851; email: schmidt-thieme@ismll.uni-hildesheim.de).

of class imbalance while [17] and [18] incorporate SVM into a boosting method.

In CSL, [20] introduced an instance-weighting method to induce cost-sensitive trees; two other methods investigated on CSL with decision trees ([22], [23]) while [24] introduced CSL with Naive Bayes. These studies introduced a test strategy which determines how unknown attributes are selected to perform test on in order to minimize the sum of the misclassification costs and test costs.

Moreover, [26] applied synthetic minority oversampling technique (SMOTE [4]) to balance the dataset first, then built the model using SVM with different costs proposed by [13]; [27] and [28] applied some common classifiers (e.g. C4.5, logistic regression, and Naive Bayes) with sampling techniques such as random undersampling, random oversampling, condensed nearest neighbor rule [8], Wilson's edited nearest neighbor rule [10], Tomek's link [9], and SMOTE.

Different to the literature, instead of focusing only on data sampling or CSL, we propose using both techniques. In addition, we do not assume a fixed cost ratio, neither set the cost ratio by inverting the ratio of prior distributions between minority and majority class; instead, we optimize the cost ratio locally.

III. Dealing with Class Imbalance

To deal with imbalanced datasets, researchers used to focus on the data level and the classifier level ([1], [2]). At the data level, the common task is the modification of the class distribution. At the classifier level, many techniques were introduced such as manipulating classifiers internally, one-class learning, ensemble learning, and CSL.

A. Modifying Class Distribution

Random oversampling (**ROS**) is a non-heuristic method [1] used to balance class distribution by randomly duplicating the minority class examples, while random undersampling (**RUS**) randomly eliminates the majority class examples.

The Condensed Nearest Neighbor Rule (**CNN**) [8] is used to find a consistent subset of examples. A subset $\hat{E} \subseteq \mathrm{E}$ is consistent with E if using the 1-nearest neighbor classifier, $\hat{E}$ correctly classifies the examples in E.

Wilson's Edited Nearest Neighbor Rule (**ENN**) [10] removes any instance with a class label different from the class of at least two of its three nearest neighbors.

Tomek's Link (**TLINK**) [9] is a method for cleaning data. Given two examples e_i and e_j belonging to different classes, $d(e_i, e_j)$ be the distance between e_i and e_j. A pair (e_i, e_j) is called a TLINK if there is no example e_l such that $d(e_i, e_l) < d(e_i, e_j)$ or $d(e_j, e_l) < d(e_i, e_j)$. If there is a TLINK between 2 examples, then either one of these is noise or both of them are borderline examples. We want to use TLINK as undersampling method, so only majority examples are removed.

One-sided selection (**OSS**) [33] is an undersampling method that first applies CNN to find a consistent subset, and then TLINK to remove noise and borderline examples.

The Synthetic Minority Oversampling Technique (**SMOTE**) is an oversampling method introduced by [4] which generates new artificial minority examples by interpolating between the existing minority examples. This method first finds the k nearest neighbors of each minority example; next, it selects a random nearest neighbor. Then a new minority class sample is created along the line segment joining a minority class sample and its nearest neighbor.

B. Cost-Sensitive Learning (CSL)

Most classifiers assume that the misclassification costs (false negative and false positive cost) are the same. In most real-world applications, this assumption is not true. For example, in customer relationship management, the cost of mailing to non-buyers is less than the cost of not mailing to the buyers [19]; or the cost of misclassifying a non-terrorist as terrorist is much lower than the cost of misclassifying an actual terrorist who carries a bomb to a flight. Another example is cancer diagnosis: misclassifying a cancer is much more serious than the false alarm since the patients could lose their life because of a late diagnosis and treatment [34]. Cost is not necessarily monetary, for examples, it can be a waste of time or even the severity of an illness [30].

This study focuses on binary classification problems; we denote the positive class ($+$ or $+1$) as the minority, and the negative class ($-$ or -1) as the majority. Let $C(i, j)$ be the cost of predicting an example belonging to class i when in fact it belongs to class j; the cost matrix is defined in Table I.

TABLE I
Cost matrix

		Predicted class	
		Positive	Negative
Actual class	Positive	$C(+,+)$	$C(-,+)$
	Negative	$C(+,-)$	$C(-,-)$

Given the cost matrix, an example x can be classified into class i with the minimum expected cost by using the Bayes risk criterion: (*conditional risk*):

$$\mathcal{H}(x) = \arg\min_i \left(\sum_{j \in \{-,+\}} P(j|x) C(i,j) \right) \quad (1)$$

where $P(j|x)$ is the posterior probability of classifying an example x as class j.

We assume that there is no cost for correct classifications, so the cost matrix can be described by the cost ratio:

$$CostRatio = C(-,+)/C(+,-) \quad (2)$$

The purpose of CSL is to build a model with minimum misclassification costs (total cost):

$$TotalCost = C(-,+) \times \#FN + C(+,-) \times \#FP \quad (3)$$

where $\#FN$ and $\#FP$ are the number of false negative and false positive examples respectively.

IV. Proposed Methods

The proposed methods are described in 4 subsections:

- We use support vector machines (SVM) as the base classifier.
- Grid search is used to determine the best hyperparameters for SVM and the resampling techniques.
- **Method 1**: Combination of Sampling techniques with CSL, called **S-CSL**.
- **Method 2**: Using CSL by Optimizing Cost Ratio Locally, called **CSL-OCRL**.

A. Support Vector Machines

Given a dataset $\mathcal{D}$ consisting of n examples (x_i, y_i), where $x_i \in \mathcal{X}$ are input features and y_i is the target class, $y_i \in \{-1, +1\}$. SVM predicts a new example x by

$$f(x) = sign\left(\sum_{i=1}^{n} \alpha_i y_i k(\mathbf{x}, \mathbf{x_i}) + b\right) \tag{4}$$

where $k(\mathbf{x}, \mathbf{x_i})$ is a kernel function, b is the bias, and α_i is determined by solving the Lagrangian optimization problem, $L_p =$

$$\frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i}^{n}\xi_i - \sum_{i}^{n}\alpha_i\{y_i(x_i.w+b) - 1 + \xi_i\} - \sum_{i}^{n}\mu_i\xi_i \tag{5}$$

where ξ_i is a slack variable, μ_i is a Lagrange multiplier, and C is a user-specified hyperparameter representing the penalty of misclassifying the training instances.

For non-linear problems, the kernel k is used to maximize margin hyperplanes. Two commonly used kernel functions are the polynomial kernel

$$k(\mathbf{x}, \mathbf{x_i}) = (\gamma \mathbf{x} \cdot \mathbf{x_i} + r)^p \tag{6}$$

and the radial basis function kernel

$$k(\mathbf{x}, \mathbf{x_i}) = e^{-\gamma\|\mathbf{x}-\mathbf{x_i}\|^2} \tag{7}$$

B. Hyperparameter Search

We have searched for the best hyperparameters C, exponent p, and γ in equations (5), (6), and (7) respectively. First, a "raw search" on the powers of two (e.g. $2^{-15} \ldots 2^{10}$ for C values) was used to identify a good region, then a "smooth search" around that region was conducted [35]. Figure 1 describes the details of this method.

Moreover, each dataset has its own structure, so the percentages of undersampling and oversampling are also different. These percentages are also treated as hyperparameters. For oversampling, we search on the percentages from 50, 100, 150, ... to a balanced distribution between the two classes. Similarly for undersampling, we also search on the percentages from 10, 20, 30, ... to a balanced distribution.

procedure HyperSearch($\mathcal{D}_{Train}, E, \delta, \lambda$)
returns the best hyperparameters Θ for eval. metric E
  $(\mathcal{D}_{LocalTrain}, \mathcal{D}_{Holdout}) \leftarrow \mathcal{D}_{Train}$ //split for 5-fold CV
  //Raw search:
  $bestC, best\gamma \leftarrow 0$
  for $i \leftarrow -15, \ldots, 10$ **do**
    for $j \leftarrow -15, \ldots, 0$ **do**
      $\gamma \leftarrow 2^j$; $C \leftarrow 2^i$
      $buildLocalSVM(\mathcal{D}_{LocalTrain}, \gamma, C)$
      $TestLocalModel(\mathcal{D}_{Holdout})$ //using metric E
      Update $bestC, best\gamma$
    end for
  end for
  //Smooth search:
  for $i \leftarrow bestC - 1, \ldots, bestC + 1, step\ \delta$ **do**
    for $j \leftarrow best\gamma - 0.1, \ldots, best\gamma + 0.1, step\ \lambda$ **do**
      $\gamma \leftarrow j$; $C \leftarrow i$
      $buildLocalSVM(\mathcal{D}_{LocalTrain}, \gamma, C)$
      $TestLocalModel(\mathcal{D}_{Holdout})$ //using metric E
      $\Theta \leftarrow C, \gamma$ //Update the best parameter values
    end for
  end for
  return Θ
end procedure

Fig. 1. Hyperparameter search for optimizing metric E with step δ for C value, and step λ for γ value in RBF kernel

C. Method 1: Combine Sampling with CSL (S-CSL)

We combine 4 resampling techniques with CSL using standard SVMs[1]. These techniques include non-heuristic under-/over-sampling (RUS, ROS) and heuristic under-/over-sampling (TLink, SMOTE). In the first step, we divide the original dataset into two separate train and test sets; then, 4 sampling techniques $\mu \in$ {RUS, TLINK, ROS, SMOTE} with different sampling percentages Φ are applied on the train set to generate new distributions; next, we perform hyperparameter search (see Figure 1) on the new training sets to determine the best parameters in terms of total costs (TC); in the next step, SVMs are built based on the best hyperparameters found. The outputs of SVM are fitted by a sigmoid function[2] to get the posterior probabilities; finally, we use the Bayes risk criterion to predict new examples in the test set. Details are described in Figure 2. The results are averaged from 5-fold cross-validation.

Most datasets do not have the cost ratios, so we assumed cost ratios from the set $\{2^2, 2^4, 2^6, 2^8\}$. The final results are reported by averaging misclassification costs for those ratios. This is also done in many other studies ([30], [21], [25]).

[1]We have used Weka's SMO, http://www.cs.waikato.ac.nz/ml/weka/

[2]The sigmoid function has 2 parameters: α and β. These values can be determined by using maximum likelihood [29], but for straightforward, we set them to 1

procedure S-CSL($\mathcal{D}, \mu, \mathcal{C}$)
Input: Dataset $\mathcal{D}$ and cost matrix $\mathcal{C}$
Output: Label for new example x^*
  $(\mathcal{D}_{Train}, \mathcal{D}_{Test}) \leftarrow \mathcal{D}$ //split for 5-fold CV
  for each Φ in {sample space percentages} **do**
    $\mathcal{D}_{Tr\mu\Phi} \leftarrow GenerateDistribution(\mathcal{D}_{Train}, \mu, \Phi)$
    $\Theta_{\mu\Phi} \leftarrow HyperSearch(\mathcal{D}_{Tr\mu\Phi}, TC, 0.25, 0.01)$
      //0.25 and 0.01 are increase-step of C and γ in RBF kernel
    $\Theta^*_{\mu\Phi} \leftarrow$ Update-best-hyperparameters for $\mathcal{D}^*_{Tr\mu\Phi}$
  end for
  //Train SVM model with parameters $\Theta^*_{\mu\Phi}$ on $\mathcal{D}^*_{Tr\mu\Phi}$

$$f(x) \leftarrow \sum_{i=1}^{n} \alpha_i y_i k(\mathbf{x}, \mathbf{x_i}) + b$$

  //Fitting a sigmoid function to SVM outputs to get the posterior probability.

$$P(j|x) \leftarrow \frac{1}{1 + e^{\alpha f(x) + \beta}}$$

  //Testing example x^* in $\mathcal{D}_{Test}$

$$\mathcal{H}(x^*) \leftarrow \arg\min_i \left(\sum_{j \in \{-1,+1\}} P(j|x^*) \mathcal{C}_{ij} \right)$$

end procedure

Fig. 2. Combination of sampling with CSL (S-CSL)

D. Method 2: CSL by Optimizing Cost Ratio Locally

In the S-CSL method, we have assumed unknown cost ratios. We tried different cost ratios and took the average result. In this section, we will introduce a method that supplies the best cost ratio to the classifier. In previous works, the cost ratio was determined by inverting the prior distributions ([7], [31]), for example, cost ratio = #majority examples/#minority examples. This choice leads to the Kolmogorov-Smirnov statistic being the performance metric [36]. Hand said that this is almost certainly inappropriate, precisely because it is made not on the basis of consideration of the relative severity of misclassifications in the presenting problem, but simply on grounds of convenience ([36], [32]). In our method, we treat this cost ratio as a hyperparameter, and locally optimize this parameter (see Figure 3). We use this kind of search because the datasets in this study are not extremely imbalanced and our preliminary experiments showed that the results are not significantly improved (in terms of the GMean metric) when using a high cost ratio. Figure 4 presents the CSL-OCRL method. This method is nearly the same as S-CSL, we just learn on the original data and optimize the cost ratio for the GMean metric[3]

[3]We used GMean as a evaluation metric in this study because previous works show that GMean is more appropriate in the case of imbalanced data ([33], [15], [17], [37]). $GMean = \sqrt{TPR \times TNR}$ [33], where TPR and TNR are the true positive rate and true negative rate.

procedure OPTIMIZECOSTRATIO($\mathcal{D}_{Train}, \Theta, \eta$)
Input: $\mathcal{D}_{Train}$, SVM parameters Θ, step length η
Outputs: the best cost ratio for GMean
  $(\mathcal{D}_{LocalTrain}, \mathcal{D}_{Val}) \leftarrow \mathcal{D}_{Train}$ ▷ split for 5-fold CV
  $ImbaRatio \leftarrow \frac{|Major|}{|Minor|}$ ▷ imbalance ratio of $\mathcal{D}_{Train}$
  $maxRatio \leftarrow ImbaRatio * 1.5$
  $curRatio \leftarrow 1.0$
  $bestGMean \leftarrow 0$
  $buildLocalModel(\mathcal{D}_{LocalTrain}, \Theta)$
  while $curRatio <= maxRatio$ **do**
    $curGMean \leftarrow testLocalModel(\mathcal{D}_{Val}, curRatio)$
    if $(curGMean > bestGMean)$ **then**
      $bestGMean \leftarrow curGMean$
      $bestCostRatio \leftarrow curRatio$
    end if
    $curRatio \leftarrow curRatio + \eta$
  end while
  return $bestCostRatio$
end procedure

Fig. 3. Locally optimize the cost ratio with step length η

V. DATASETS

We have experimented on 18 imbalanced datasets from the UCI repository[4], as described in Table II. Some multi-class datasets are converted to binary datasets using the one-class-versus-rest scheme. The imbalance ratio ranges from 1.77 (lowest) to 64.03 (highest) between the majority and minority examples. Since each dataset is generated by 4 different sampling techniques, we have actually experimented on 90 "datasets", including the original ones.

TABLE II
DATASETS

Dataset	#Examples	#Attributes	#Minority	Imba. Ratio
Abalone	4,177	9	391	9.68
Allbp	2,800	30	133	20.05
Allhyper	3,772	30	102	35.98
Allrep	3,772	30	124	29.45
Ann	7,200	22	166	42.37
Anneal	898	39	40	21.45
Breastcancer	699	11	241	1.90
Diabetes	768	9	268	1.86
Dis	3,772	30	58	64.03
Heartdisease	294	14	106	1.77
Hepatitis	155	20	32	3.84
Hypothyroid	3,163	26	151	19.95
Nursery	12,960	9	328	38.51
Pima-Indian	768	9	268	1.87
Sick	2,800	30	171	15.37
Spectheart	267	23	55	3.85
Transfusion	748	5	178	3.20
Wpbc	198	34	47	3.21

[4]http://archive.ics.uci.edu/ml/

procedure CSL-OCRL($\mathcal{D}$)

Input: Dataset $\mathcal{D}$

Output: Label for new example x^*

$(\mathcal{D}_{Train}, \mathcal{D}_{Test}) \leftarrow \mathcal{D}$ //split for 5-fold CV

$\Theta \leftarrow HyperSearch(\mathcal{D}_{Train}, GMean, 0.25, 0.01)$

//Optimize locally with increase-step 0.25 for cost ratio

$$C^*(i,j) \leftarrow OptimizeCostRatio(\mathcal{D}_{Train}, \Theta, 0.25)$$

//Train SVM model with parameters Θ on $\mathcal{D}_{Train}$

$$f(x) \leftarrow \sum_{i=1}^{n} \alpha_i y_i k(\mathbf{x}, \mathbf{x_i}) + b$$

//Fitting a sigmoid function to SVM outputs to get the posterior probability:

$$P(j|x) \leftarrow \frac{1}{1 + e^{\alpha f(x) + \beta}}$$

//Testing example x^* in $\mathcal{D}_{Test}$:

$$\mathcal{H}(x^*) \leftarrow \arg\min_i \left(\sum_{j \in \{-1,+1\}} P(j|x^*) C^*(i,j) \right)$$

end procedure

Fig. 4. CSL by Optimizing Cost Ratio Locally

VI. Experimental Results

A. Results of Method 1 (S-CSL)

The sampling scheme is <Sampling method> <Percentage>. For example, SM100 and ROS200 denote SMOTE and random oversampling with 100% and 200%, respectively. We have implemented 4 combinations and compared them with three other CSL methods, which are MetaCost ([30]), CSL on original data ([19], denoted by CSL), and CSL by instance weighting ([20], [38], denoted by CSW). Figure 5 shows the relationship between cost ratios and total costs of these methods in five typical results. One can see clearly that when the cost ratio increases, our methods reduce the total cost significantly. This consolidates the results of our initial study [39]. CSL as a meta-learning method and the internal classifier (SVM in this case) are still impacted by the class imbalance problem. CSL can work better if it is supplied by a re-balanced dataset.

Table III compares the results of S-CSL with other methods in term of average costs. For each dataset, when comparing the last four columns (S-CSL) with the other methods, we can see that the average misclassification costs are reduced after re-sampling in most cases. For each row in the table, the **bold number** denotes the best result and the *italic number* describes our combination better than MetaCost. We also report the percentage for the sampling methods, and the imbalance ratio after resampling for each dataset. The combination of RUS with CSL (RUS-CSL) works better than the remaining combinations. In addition, RUS-CSL is always works better than MetaCost, CSL, and

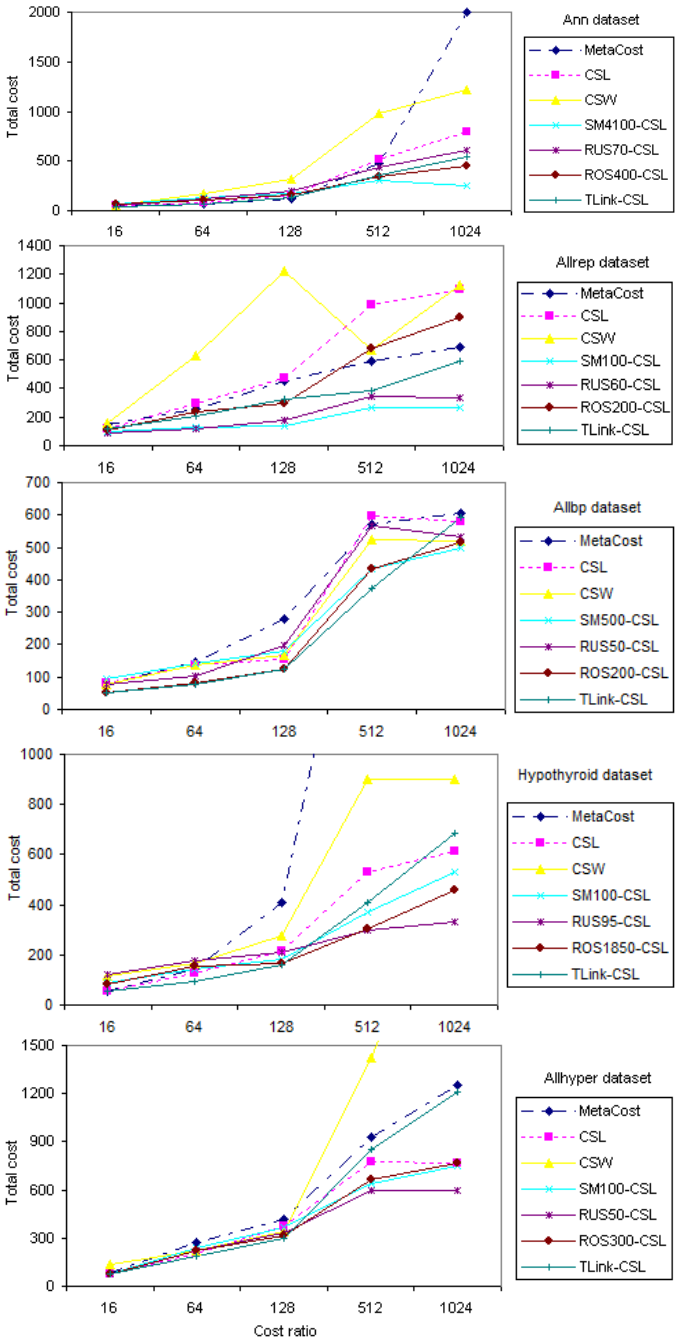

Fig. 5. Cost ratio and total cost relationship for 5 typical results

CSW (except for the Dis dataset). The last row in the table summarizes the comparison results of each combination with 3 other methods.

Moreover, when observing the imbalance ratio before and after sampling, the results show that not only class imbalance, but also noise, borderline examples, and class overlapping degrade the classifier performance. These problems have also been reported by [9], [33], [40].

TABLE III

EXPERIMENTAL RESULTS - AVERAGE COSTS FOR METHOD 1: S-CSL

Dataset	SVM	MetaCost	CSL	CSW	***S-CSL***			
					ROS-CSL	RUS-CSL	SMOTE-CSL	TLink-CSL
Abalone	6632.00	547.00	486.25	496.90	*486.80*	***460.75***	*486.05*	*469.50*
(%Sampling, Imbalance ratio)	(-, 9.68)				(100, 4.85)	(30, 6.79)	(100, 4.85)	(-, 8.97)
Allbp	1771.00	364.50	342.80	373.65	*337.75*	***290.50***	*320.30*	*302.10*
	(-, 20.05)				(300, 5.03)	(20, 16.11)	(500, 3.35)	(-, 19.50)
Allhyper	887.20	373.05	227.00	259.45	*222.30*	*172.60*	*193.20*	***167.90***
	(-, 35.98)				(400, 7.24)	(50, 18.12)	(100, 18.12)	(-, 35.01)
Allrep	1320.00	93.65	112.35	399.25	***84.40***	*84.55*	135.65	111.00
	(-, 29.45)				(300, 7.36)	(80, 5.89)	(2800, 1.01)	(-, 28.79)
Ann	787.00	134.80	131.45	182.05	140.45	***98.10***	*131.05*	*113.40*
	(-, 42.37)				(100, 21.31)	(50, 21.31)	(200, 14.21)	(-, 42.10)
Anneal	154.20	57.40	55.15	55.10	*51.20*	*53.15*	***38.15***	69.30
	(-, 21.45)				(400, 4.28)	(20, 17.15)	(200, 7.14)	(-, 21.25)
Breastcancer	206.20	24.30	25.80	23.60	*14.90*	*14.45*	*24.70*	***13.15***
	(-, 1.90)				(70, 1.12)	(30, 1.33)	(50, 1.27)	(-, 1.85)
Diabetes	1916.80	91.65	89.90	172.45	*85.80*	***86.10***	*86.25*	*90.40*
	(-, 1.86)				(60, 1.16)	(20, 1.49)	(60, 1.16)	(-, 1.52)
Dis	958.80	350.65	422.85	**304.80**	*326.25*	*337.65*	356.05	371.30
	(-, 64.03)				(6000, 1.05)	(50, 32.30)	(500, 10.76)	(-, 63.97)
Heartdisease	598.40	34.10	32.50	195.25	***31.15***	*31.45*	*31.70*	36.30
	(-, 1.77)				(20, 1.51)	(20, 1.44)	(30, 1.38)	(-, 1.28)
Hepatitis	273.60	18.80	19.20	76.55	***18.15***	*18.70*	20.90	19.85
	(-, 3.84)				(50, 2.67)	(30, 2.80)	(100, 1.98)	(-, 3.40)
Hypothyroid	700.00	124.30	140.10	197.20	156.75	*103.45*	***101.50***	*110.55*
	(-, 19.95)				(100, 10.04)	(20, 16.06)	(100, 10.04)	(-, 19.70)
Nursery	1317.8	115.75	108.45	42.75	*42.80*	*37.95*	***26.40***	*35.00*
	(-, 38.51)				(100, 19.28)	(70, 11.57)	(300, 9.64)	(-, 36.56)
Pima-Indian	2017.6	96.3	97.05	128.8	*88.25*	***87.95***	*88.85*	105.00
	(-, 1.87)				(50, 1.24)	(20, 1.49)	(70, 1.10)	(-, 1.50)
Sick	1316.40	254.20	207.00	437.80	*253.35*	*206.65*	***201.20***	256.80
	(-, 15.37)				(200, 5.15)	(20, 12.38)	(100, 7.73)	(-, 15.12)
Spectheart	462.00	34.05	36.75	161.35	40.50	***32.25***	*33.95*	*33.30*
	(-, 3.85)				(200, 1.28)	(50, 1.93)	(50, 2.56)	(-, 2.95)
Transfusion	2840.40	108.6	109.8	105.3	*106.65*	***103.85***	*106.45*	*106.05*
	(-, 3.20)				(50, 2.14)	(10, 2.89)	(50, 2.14)	(-, 2.53)
Wpbc	463.40	33.00	29.90	74.55	*28.40*	***26.35***	*28.05*	*28.55*
	(-, 3.21)				(100, 1.63)	(20, 2.62)	(100, 1.63)	(-, 2.65)
Number of times S-CSL works better than MetaCost/CSL/CSW (out of 18)					*15/13/15*	*18/18/17*	*14/16/15*	*12/12/14*

Note: The lower the cost, the better the model

B. Results of Method 2 - CSL-OCRL

Table IV compares the result of CSL-OCRL with other CSL and meta-learning methods, which are CSL by instance weighting (CSW) ([20], [38]), MetaCost ([30]), Threshold-Selector ([38]), and AdaBoost-CSL ([41], [42]). We use a paired t-test with significance level 0.05. We use the CSL-OCRL method as a baseline and compare the other methods against it. The bold numbers present the best results among these methods. One can see clearly that CSL-OCRL is almost always equal to, or better than other methods.

VII. CONCLUSIONS

When learning from imbalanced data, the classifiers are usually overwhelmed by the majority class, so the minority class examples tend to be misclassified. Along with sampling techniques and modifying the classifiers internally, CSL is also an important approach because it takes into account different misclassification costs for false negatives and false positives.

In this study, we have proposed two simple methods to deal with class imbalance. A key feature of our methods is that we do not need to change the classifiers internally, so they are easy to implement. The first method combines sampling techniques with CSL to reduce the total misclassification costs of the model. Experimental results show that in most cases, the misclassification costs are reduced by using this combination. The second method (CSL-OCRL) optimizes the cost ratio locally and applies this ratio to train the full model. The GMean results show that CSL-OCRL usually performs at least as good as the other methods, and is significantly better than some methods in certain cases.

In the future, we will study how to combine CSL with feature selection methods, and investigate the influence of class imbalance on large datasets.

ACKNOWLEDGMENTS

The first author was funded by the "Teaching and Research Innovation Grant" Project of Can Tho University, Vietnam.

TABLE IV

EXPERIMENTAL RESULTS - GMEAN FOR METHOD 2: CSL-OCRL

Dataset	CSL-OCRL	CSW	MetaCost	ThresholdSelector	AdaBoost-CSL
abalone	0.779±0.015	0.784±0.006	0.779±0.020	0.738±0.023 •	0.798±0.017
allbp	**0.870**±0.032	0.823±0.055	0.865±0.028	0.722±0.058 •	0.797±0.074
allhyper	**0.895**±0.042	0.841±0.084	0.893±0.073	0.776±0.021 •	0.791±0.067 •
allrep	**0.886**±0.031	0.789±0.061 •	0.874±0.033	0.736±0.075 •	0.780±0.065 •
ann	0.949±0.033	0.955±0.041	0.970±0.011	0.882±0.049 •	0.922±0.049
anneal	**0.968**±0.057	0.946±0.055	0.962±0.057 •	0.743±0.419	0.932±0.068
breastcancer	**0.969**±0.016	0.968±0.019	0.965±0.011	0.965±0.011	0.944±0.019 •
diabetes	**0.760**±0.043	0.746±0.048 •	0.705±0.046 •	0.610±0.072 •	0.713±0.055 •
dis	**0.739**±0.081	0.641±0.109 •	0.738±0.184	0.656±0.155	0.545±0.172 •
heartdisease	**0.828**±0.064	0.818±0.067	0.776±0.049	0.796±0.062	0.781±0.091
hepatitis	0.755±0.061	0.747±0.071	0.725±0.075	0.764±0.082	0.763±0.058
hypothyroid	0.899±0.044	0.856±0.038 •	0.927±0.034 ○	0.799±0.103 •	0.818±0.060 •
nursery	**1.000**±0.000	1.000±0.000	0.995±0.000 •	0.853±0.295	0.998±0.003
pima	**0.747**±0.050	0.737±0.031	0.697±0.052	0.727±0.038	0.710±0.037
sick	**0.912**±0.029	0.870±0.054	0.912±0.033	0.852±0.079	0.863±0.044
spectheart	**0.772**±0.037	0.732±0.082	0.730±0.076	0.756±0.055	0.739±0.117
transfusion	0.678±0.027	0.682±0.021	0.661±0.008	0.680±0.018	0.693±0.028
wpbc	**0.683**±0.056	0.619±0.194	0.680±0.084	0.257±0.269 •	0.678±0.116
Average	0.838	0.809	0.826	0.740	0.793
win/tie/lose	base	0/14/4	1/14/3	0/10/8	0/12/6

○, • *statistically significant improvement or degradation, level=0.05*

REFERENCES

[1] N. V. Chawla, N. Japkowicz, and A. Kotcz, "Editorial: special issue on learning from imbalanced data sets," *SIGKDD Explorations*, vol. 6, no. 1, pp. 1–6, 2004.

[2] H. He and E. A. Garcia, "Learning from imbalanced data," *IEEE Transactions on Knowledge and Data Engineering*, vol. 21, no. 9, pp. 1263–1284, September 2009.

[3] Q. Yang and X. Wu, "10 challenging problems in data mining research," *International Journal of Information Technology and Decision Making*, vol. 5, no. 4, pp. 597–604, 2006.

[4] N. V. Chawla, K. Bowyer, L. Hall, and W. P. Kegelmeyer, "SMOTE: synthetic minority over-sampling technique," *Journal of AI Research*, vol. 16, pp. 321–357, 2002.

[5] A. Nickerson, N. Japkowicz, and E. Millos, "Using unsupervised learning to guide resampling in imbalanced data sets," in *Proceedings of the Eighth International Workshop on AI and Statitsics*, 2001, pp. 261–265.

[6] P. Li, P.-L. Qiao, and Y.-C. Liu, "A hybrid re-sampling method for SVM learning from imbalanced data sets," in *Proceedings of the 2008 Fifth IC on Fuzzy Systems and Knowledge Discovery*. Washington, DC, USA: IEEE Computer Society, 2008, pp. 65–69.

[7] B. Raskutti and A. Kowalczyk, "Extreme re-balancing for SVMs: a case study," *SIGKDD Explorations*, vol. 6, no. 1, pp. 60–69, 2004.

[8] P. E. Hart, "The condensed nearest neighbor rule," *IEEE Trans. Inf. Theory*, no. 16, pp. 515–516, 1968.

[9] I. Tomek, "Two modifications of CNN," *IEEE Transactions on Systems Man and Communications SMC-6*, pp. 769–772, 1976.

[10] D. L. Wilson, "Asymptotic properties of nearest neighbor rules using edited data," *IEEE Transactions on Systems, Man and Cybernetics*, no. 3, 1972.

[11] X. wen Chen, B. Gerlach, and D. Casasent, "Pruning support vectors for imbalanced data classification," *proceeding of IEEE International Joint Conference on Neural Networks*, vol. 3, pp. 1883–1888, 2005.

[12] S. Lessmann, "Solving imbalanced classification problems with support vector machines," in *International Conference on Artificial Intelligence*, 2004, pp. 214–220.

[13] K. Veropoulos, C. Campbell, and N. Cristianini, "Controlling the sensitivity of support vector machines," in *Proceedings of the IJCAI*, 1999, pp. 55–60.

[14] R. Yan, Y. Liu, R. Jin, and A. Hauptmann, "On predicting rare classes with SVM ensembles in scene classification," in *ICASSP*, 2003, pp. 21–24.

[15] X.-Y. Liu, J. Wu, and Z.-H. Zhou, "Exploratory under-sampling for class-imbalance learning," *IEEE Trans. on Syst, Man, and Cyber. Part B*, pp. 539–550, 2009.

[16] Y. Tang, Y. Zhang, N. Chawla, and S. Krasser, "SVMs modeling for highly imbalanced classification." *IEEE Trans Syst Man Cybern B Cybern*, vol. 39, no. 1, pp. 281–8, 2009.

[17] B. Wang and N. Japkowicz, "Boosting support vector machines for imbalanced data sets," *Knowledge and Information Systems*, 2009.

[18] X. Li, L. Wang, and E. Sung, "AdaBoost with SVM-based component classifiers," *Eng. Appl. Artif. Intell.*, vol. 21, no. 5, pp. 785–795, 2008.

[19] C. Elkan, "The foundations of cost-senstive learning," *17th International Joint Conference on Artificial Intelligence*, pp. 973–978, 2001.

[20] K. M. Ting, "Inducing cost-sensitive trees via instance weighting," in *The 2nd European Symposium on Principles of KDD*. Springer-Verlag, 1998, pp. 139–147.

[21] X.-Y. Liu and Z.-H. Zhou, "The influence of class imbalance on cost-sensitive learning: An empirical study," in *Proceedings of the Sixth ICDM*. Washington, DC, USA: IEEE Computer Society, 2006, pp. 970–974.

[22] V. S. Sheng, C. X. Ling, A. Ni, and S. Zhang, "Cost-sensitive test strategies," in *Conference on Artificial Intelligence (AAAI)*, 2006.

[23] S. Sheng, C. X. Ling, and Q. Yang, "Simple test strategies for cost-sensitive decision trees," in *ECML*, 2005, pp. 365–376.

[24] X. Chai, L. Deng, Q. Yang, and C. X. Ling, "Test-cost sensitive naive bayes classification," in *International Conference on Data Mining*, 2004, pp. 51–58.

[25] K. M. Ting, "A study on the effect of class distribution using cost-sensitive learning," in *Discovery Science*, 2002, pp. 98–112.

[26] R. Akbani, S. Kwek, and N. Japkowicz, "Applying support vector machines to imbalanced datasets," in *Proceedings of the 15th ECML*, 2004, pp. 39–50.

[27] G. E. A. P. A. Batista, R. C. Prati, and M. C. Monard, "A study of the behavior of several methods for balancing machine learning training data," *SIGKDD Explorations*, vol. 6, no. 1, pp. 20–26, 2004.

[28] J. Van Hulse, T. M. Khoshgoftaar, and A. Napolitano, "Experimental perspectives on learning from imbalanced data," in *Proceedings of 24th ICML*. ACM, 2007, pp. 935–942.

[29] J. C. Platt, "Probabilistic outputs for SVM and comparisons to regularized likelihood methods," in *Advances in Large Margin Classifiers*. MIT Press, 1999, pp. 61–74.

[30] P. Domingos, "Metacost: A general method for making classifiers cost-sensitive," *5th ACM SIGKDD*, pp. 155–164, 1999.

[31] D. D. Margineantu, "When does imbalanced data require more than cost-sensitive learning?" in *Workshop on Learning from Imbalanced Data, Artificial Intelligence (AAAI-2000)*, 2000.

[32] D. J. Hand, "Classifier technology and the illusion of progress," *Statistical Science*, vol. 21, no. 1, pp. 1–14, Jun 2006.

[33] M. Kubat and S. Matwin, "Addressing the curse of imbalanced training sets: One-sided selection," in *Proceedings of the 14th ICML*. Morgan Kaufmann, 1997, pp. 179–186.

[34] V. S. Sheng and C. X. Ling, "Thresholding for making classifiers cost-sensitive," in *Conference on Artificial Intelligence (AAAI)*, 2006.

[35] C. W. Hsu, C. C. Chang, and C. J. Lin, *A practical guide to support vector classification*, Department of Computer Science and Information Engineering, National Taiwan University, 2003.

[36] D. J. Hand, "Measuring classifier performance: A coherent alternative to the area under the ROC curve," *Machine Learning*, vol. 77, no. 1, pp. 103–123, October 2009.

[37] S. Hido and H. Kashima, "Roughly balanced bagging for imbalanced data." in *Proceedings of the SIAM International Conference on Data Mining*, 2008, pp. 143–152.

[38] I. H. Witten and E. Frank, *Data Mining: Practical machine learning tools and techniques*, 2nd ed. Morgan Kaufmann, San Francisco, 2005.

[39] N. Thai-Nghe, A. Busche, and L. Schmidt-Thieme, "Improving academic performance prediction by dealing with class imbalance," *in Proceeding of 9th IEEE International Conference on Intelligent Systems Design and Applications (ISDA09)*, 2009.

[40] R. C. Prati, G. E. A. P. A. Batista, and M. C. Monard, "Class imbalances versus class overlapping: An analysis of a learning system behavior," in *MICAI04: Advances in AI, Mexico*, 2004, pp. 312–321.

[41] Y. Freund and R. E. Schapire, "Experiments with a new boosting algorithm," *International Conference on Machine Learning*, pp. 148–156, 1996.

[42] R. E. Schapire, Y. Singer, and A. Singhal, "Boosting and rocchio applied to text filtering." in *Proceedings of SIGIR-98, 21st ACM International Conference on Research and Development in Information Retrieval*. ACM Press, New York, US, 1998, pp. 215–223.