A Measure optimized cost-sensitive learning framework for imbalanced data classification

Peng Cao
Key Laboratory of Medical Image Computing of Ministry of Education, Northeastern University, China
University of Alberta, Canada

Osmar Zaiane
University of Alberta, Canada

Dazhe Zhao
Key Laboratory of Medical Image Computing of Ministry of Education, Northeastern University, China

ABSTRACT
Class imbalance is one of the challenging problems for machine learning in many real-world applications. Many methods have been proposed to address and attempt to solve the problem, including sampling and cost-sensitive learning. The latter has attracted significant attention in recent years to solve the problem, but it is difficult to determine the precise misclassification costs in practice. There are also other factors that influence the performance of the classification including the input feature subset and the intrinsic parameters of the classifier. This paper presents an effective wrapper framework incorporating the evaluation measure (AUC and G-mean) into the objective function of cost sensitive learning directly for improve the performance of classification, by simultaneously optimizing the best pair of feature subset, intrinsic parameters and misclassification cost parameter. The optimization is based on Particle Swarm Optimization (PSO).We use two different common methods, support vector machine and feed forward neural networks to evaluate our proposed framework. Experimental results on various standard benchmark datasets with different ratios of imbalance and a real-world problem show that the proposed method is effective in comparison with commonly used sampling techniques.

INTRODUCTION
Recently, the class imbalance problem has been recognized as a crucial problem in machine learning and data mining (Chawla, Japkowicz &Kolcz, 2004; Kotsiantis, Kanellopoulos& Pintelas, 2006; He &Garcia, 2009; He & Ma, 2013). This issue of imbalanced data occurs when the training data is not evenly distributed among classes. This problem is also especially critical in many real applications, such as credit card fraud detection when fraudulent cases are rare or medical diagnoses where normal cases are the majority, and it is growing in importance and has been identified as one of the 10 main challenges of Data Mining (Yang, 2006). In these cases, standard classifiers generally perform poorly. Classifiers usually tend to be overwhelmed by the majority class and ignore the minority class examples. Most classifiers assume an even distribution of examples among classes and assume an equal misclassification cost. Moreover, classifiers are typically designed to maximize accuracy, which is not a good metric to evaluate effectiveness in the case of imbalanced training data. Therefore, we need to improve traditional algorithms so as to handle imbalanced data and choose other metrics to measure performance instead of accuracy. We focus our study on imbalanced datasets with binary classes.

Much work has been done in addressing the class imbalance problem. These methods can be grouped in two categories: the data perspective and the algorithm perspective (He &Garcia 2009). The methods with the data perspective re-balance the class distribution by re-sampling the data space either

randomly or deterministically (Chawla, Bowyer, Hall & Kegelmeyer, 2002; Chawla, Lazarevic, Hall & Bowyer, 2003; Chawla, Cieslak, Hall & Joshi, 2008; Barua, Monirul Islam, Yao & Murase, 2013; Galar, Fernández, Barrenechea & Herrera, 2013). The main disadvantage of re-sampling techniques are that they may cause loss of important information or the model overfitting, since that they change the original data distribution. In addition, the performance of sampling can vary significantly depending upon the data available.

Cost-sensitive learning is one of the most important topics in machine learning and data mining, and attracted high attention in recent years (Akbani, Kwek & Japkowicz, 2004; Ling & Sheng, 2008; Zhou & Liu, 2006). Cost-sensitive learning methods consider the costs associated with misclassifying examples, and try to learn more characteristics of samples with the minority class by setting a high cost to the misclassification of a minority class sample. It has been shown that the problem of learning from imbalanced datasets and the problem of learning when costs are unequal and unknown can be handled in the same manner even though these problems are not exactly the same (Maloof, 2003). Cost-sensitive learning does not modify the data distribution, and is generally more consistent in terms of performance than the sampling techniques (Chris, Taghi, Jason & Amri, 2008; Weiss, McCarthy & Zabar, 2007).

There are two challenges with respect to the training of cost sensitive classifier. The misclassification costs play a crucial role in the construction of a cost sensitive learning model for achieving expected classification results. However, in many contexts of imbalanced dataset, the misclassification costs cannot be determined. Beside the cost, the feature set and intrinsic parameters of some sophisticated classifiers also influence the classification performance. The imbalanced data distribution is often accompanied by high dimensionality in real-world data sets such as text classification and bioinformatics (Lusa, 2013; Van Hulse, Khoshgoftaar, Napolitano & Wald, 2009; Zheng, Wu & Srihari, 2004). Therefore, high-dimensionality poses additional challenges when dealing with class-imbalanced prediction. Optimal feature selection can concurrently achieve good accuracy and dimensionality reduction. The proper intrinsic parameter setting of classifiers, such as regularization cost parameter and the kernel function parameter for SVM, and the structure parameters (i.e. number of hidden layers and their nodes) for neural network, can improve the classification performance. Moreover, these factors including the feature subset choice influence each other, obtaining the optimal factors of imbalanced data learning methods must occur simultaneously. This is the first challenge.

The other is the gap between the measure of evaluation and the objective of training on the imbalanced data (Li, Tsang, Zhou, 2012; Yuan & Liu, 2011). Indeed, for evaluating the performance of a cost-sensitive classifier on a skewed data set, the overall accuracy is irrelevant. It is common to employ other evaluation measures to monitor the balanced classification ability, such as G-mean and AUC. However, these cost-sensitive classifiers measured by imbalanced evaluation are not trained and updated with the objective of the imbalanced evaluation. To achieve good prediction performance, learning algorithms should train classifiers by optimizing the concerned performance measures.

In order to solve the challenges above, we design a novel framework for training a cost-sensitive neural network driven by the imbalanced evaluation criteria. The training scheme can bridge the gap between the training and the evaluation of cost-sensitive learning, and it can learn the optimal factors associated with the cost-sensitive classifier automatically under the guidance of the performance metrics. The search space is expanded exponentially as the class number increases. Moreover the factors to be searched are mixture including continuous and discrete variables. The significance of the scheme has two questions to fix: how to optimize these factors simultaneously; and using what evaluation criteria for guiding their optimization. These two issues are our key steps for improving the cost sensitive learning in the context of the class imbalance problem without cost information. Our main contributions in this paper are centered around the questions above.

The contributions of this work can be listed as follows:

1) Optimizing the factors (ratio misclassification cost, feature set and intrinsic parameters of classifier) simultaneously for improving the performance of cost-sensitive learning.
2) Imbalanced data classification is commonly evaluated by measures such as G-mean and AUC instead of accuracy. However, for many classifiers, the learning process is still largely driven

by error based objective functions. We use the measure directly to train the classifier and discover the optimal parameter, ratio cost and feature subset based on different evaluation functions like the G-mean or AUC. Different metrics can reflect different aspect performance of classifiers.

3) Showing versatility of our proposed framework, we present two different cost-sensitive learning schemes: one based on SVM as a direct method and one based on neural networks as meta learning method.

This chapter will be organized as follows. The basic concepts that are necessary to understand the issues addressed in this paper are described in Section 1, including imbalanced data learning, cost sensitive learning methods and particle swarm optimization. Then our proposed measure optimized framework is presented in Section 2. Section 3 details the experimental results comparing our approaches to other methods proposed in the literature for imbalanced data. Section 4 concludes with general remarks.

BACKGROUND

Imbalanced data

A common problem faced in data mining is dealing with class imbalance. A dataset is said to be imbalanced if one class (called the majority, or negative class) vastly out-numbers the other (called the minority, or positive class). The class imbalance problem is only said to exist when the positive class is the class of interest. This is due to the fact that if the positive, minority, class is not of interest (i.e., it has no effect on the choice made), then it can be safely ignored. In most practical applications (e.g., loan recommendation, fraud prevention, spam detection, intrusion detection, species modeling, long term epidemiological studies, climate data analysis, etc.), however, the minority classis the class of interest, and therefore the class imbalance problem must be addressed.

Cost sensitive learning

The significant shortcomings with the re-sampling approach are the optimal class distribution is always unknown and the criterion in selecting instances is uncertain; furthermore under-sampling may reduce information loss and over-sampling may lead to overfitting or overgeneralization for model constructed. The cost-sensitive learning technique takes misclassification costs into account during the model construction, and does not modify the imbalanced data distribution directly. Assigning distinct costs to the training examples seems to be the most effective approach of class imbalanced data problem.

The problem of imbalanced data is often associated with asymmetric costs of misclassifying instances of different classes. Medical diagnosis is a prominent example: misclassifying a cancer patient (false negative) may lead to death, while misclassifying a healthy patient (false positive) would lead to expenses associated with unnecessary biopsy and psychological problems. Datasets with different class distributions lead to the effect that conventional machine learning methods are typically biased towards the larger class in the training data.

The cost matrix contains the misclassification information: $C(+,+)$ and $C(-,-)$ are zeros, while $C(-,+)$ and $C(+,-)$ are important cost information to be determined. Moreover, $C(-,+)$ (i.e. when a minority instance is put in a majority class) should be bigger than $C(+,-)$, see Table 1. Table 1 illustrates the confusion matrix and cost matrix. The confusion matrix contains information about actual and predicted classifications done by a classification system.

Table 1. The data sets used for experimentation Confusion and Cost Matrix

		Actual class	
		Positive class	Negative class
Predicted class	Positive class	True positive (TP) $C(+,+)$	False positive (FP) $C(+,-)$
	Negative class	False negative (FN) $C(-,+)$	True negative (TN) $C(-,-)$

This study focuses on binary classification, we denote the positive class (+) as the minority and the negative class (-) as the majority. Let *C(i, j)* be the cost of predicting an instance belonging to class *i* when in fact it belongs to class *j*.

Cost-sensitive learning can be classified into two categories: direct methods and wrappers (Ling & Sheng, 2008). Direct methods are cost-sensitive classifiers in themselves, such as cost sensitive SVM; while wrappers convert any existing cost-insensitive (or cost-blind or cost-agnostic) classifiers into cost-sensitive ones. Wrappers are also called cost-sensitive meta-learning methods.

To show versatility of our method, we present two different cost-sensitive learning schemes: one based on SVM as a direct method and one based on neural networks as meta learning method.

CS-SVM

Support Vector Machines (SVM), which has strong mathematical foundations based on statistical learning theory, has been successfully adopted in various classification applications. SVM maximizes a margin in a hyperplane separating classes, and can be formulated as the following quadratic program:

$$\begin{aligned} &Min\ \ \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n}\xi_i \\ &s.t.\ \ y_i[(\mathbf{w}^T \bullet \mathbf{x_i}) + b] \geq 1 - \xi_i \ \ i = 1, \cdots, n \\ &\qquad \xi_i \geq 0 \end{aligned} \qquad (1)$$

where C ≥ 0 is a regularization parameter that controls the trade-off between minimizing the errors

and maximizing the margin. However, it is overwhelmed by the majority class instances in the case of imbalanced datasets because the objective of regular SVM is to maximize the accuracy, and not purposely to minimize the misclassification cost. The above formulation in Equation 3 implicitly penalizes errors in both classes equally. There may be different costs associated with the two different kinds of errors, making errors on positive examples costlier than errors on negative examples.

SVM have been extensively studied and have shown remarkable success in many applications. However, the success of standard SVM is very limited when applied to the problem of learning from imbalanced datasets. The cost-sensitive version of SVM (CS-SVM or 2C-SVM) (Veropoulos, Campbell & Cristianini, 1999) by assigning different misclassification costs is a good solution to address the above problem. Various proposals of cost-sensitive SVM were made using different error costs for the positive (C+) and negative (C-) classes. CS-SVM is formulated as follows:

$$\begin{aligned} &Min\ \ \frac{1}{2}\|\boldsymbol{w}\|^2 + C_+\sum_{i:y_i=+1}\xi_i + C_-\sum_{j:y_j=-1}\xi_j \\ &s.t.\ \ y_i[(\mathbf{w}^T \bullet \mathbf{x_i}) + b] \geq 1 - \xi_i \ \ i = 1, \cdots, n \\ &\qquad \xi_i \geq 0 \end{aligned} \qquad (2)$$

where the C_+, or the $C(-,+)$, is the higher misclassification cost of the positive class, which is the primary interest, while C_-, or the $C(+,-)$, is the lower misclassification cost of the negative class. Using the different error cost for the positive and negative classes, the hyperplane could be pushed away from the positive instances. In this article, we fix $C_- = C$ and $C_+ = C \times C_{rf}$, where C_{rf} is the ratio misclassification cost factor.

In general, the Radial Basis Function (RBF kernel) is a reasonable first choice for the classification of the nonlinear datasets, as it has fewer parameters (γ). For the cost information, Veropoulos el at

(Veropoulos, Campbell & Cristianini, 1999) have not suggested any guidelines for deciding what the relative ratios of the positive to negative cost factors should be.

CS-NN

The standard neural network is cost insensitive. In standard neural network classifiers, the class returned is *C** by comparing the probability of each class directly for each instance *x* according to **Eq.**(3).

$$C^* = \underset{C\in\{1,...,M\}}{argmax}(p_1(C_1 \mid x),..., p_M(C_M \mid x)) \qquad (3)$$

where P_i denotes the probability value of each class from the neural network, $\sum_{i=1}^{M} P_i = 1$ and $0 \le P_i \le 1$. M is the number of the class.

Many approaches have been developed in the past few years in making the traditional cost-insensitive neural network classification algorithm into cost-sensitive (Kukar & Kononenko, 1998; Zhou & Liu, 2006). The probabilities generated by a standard neural network are biased in the imbalanced data distribution, adjusting the decision threshold moves the output threshold toward inexpensive class such that instances with high costs become harder to be misclassified (Ling & Sheng, 2008). The idea is based on the classifier producing probability predictions rather than classification labels. Results suggest that threshold-moving, replacing the probability a sample belongs to a certain class with the altered probability, which takes into account the costs of misclassification, is found to be a relatively good choice in training CS-NN (Zhou & Liu, 2006). This method uses the training set to train a neural network, and the cost sensitivity strategy is introduced in the test phase. Given a certain cost matrix, the CS-NN with threshold-moving return the class *C**, which is computed by injecting the cost according to **Eq.**(4).

$$\begin{aligned} C^* &= \underset{C}{argmax} \frac{1}{\sum_{i=1}^{M} p_i^*(C_i \mid x)} \left\{ p_1^*(C_1 \mid x),..., p_M^*(C_M \mid x) \right\} \\ &= \underset{C}{argmax} \frac{1}{\sum_{i=1}^{M} cost(C_i)\, p(C_i \mid x)} \left\{ cost(C_1)\, p(C_1 \mid x),..., cost(C_M) \times p(C_M \mid x) \right\} \end{aligned} \qquad (4)$$

where $Cost(C_i)$ denotes the cost of misclassifying instance of class *i*. P_i^* denotes the class probabilities from the neural network combined with misclassification cost.

When *M* is 2 (binary class), the classifier will classify an instance *x* into minority class if and only if:

$$p(+ \mid x)C(-,+) > p(- \mid x)C(+,-) \qquad (5)$$

which is equivalent to

$$\frac{p(+ \mid x)}{p(- \mid x)} > \frac{C(+,-)}{C(-,+)} = C_{rf} \qquad (6)$$

It predicts the class by setting a probability threshold dependent on the ratio misclassification cost. Therefore, the final decisions are decide by the misclassification cost specified and probability estimate learned. In the normal classification without considering the cost, the C_{rf} is 1, the decision threshold is 0.5, that means both of the two classes have the same weight. In the cost sensitive context, we need to improve the recognition ability of minority class. Unlike the SVM the ratio cost C_{rf} is used in the training phrase; it is introduced in the validation phase after obtaining a common neural network in the training phrase. When validating, we observe a probability estimate *p* belonging to positive class on a testing instance, the instance is labeled as the positive class or negative class according to C_{rf} through (6).

Particle swarm optimization

Swarm Intelligence (SI), an artificial intelligence technique for machine learning, is a research branch that models the population of interacting agents or swarms that are able to self-organize. SI has recently emerged as a practical research topic and has successfully been applied to a number of real world problems (Martens, Baesens & Fawcett, 2011). The popularity of swarm intelligence has also instigated the development of numerous data mining algorithms.

Particle swarm optimization (PSO) is a population-based global stochastic search method attributed to Kennedy and Eberhart to simulate social behavior (Kennedy & Eberhart, 1995). Compared to Genetic Algorithms (GA), the advantages of PSO are that it is easy to implement and has fewer control parameters to adjust. Many studies have shown than PSO has the same effectiveness but is more efficient. PSO optimizes an objective function by a population-based search. The population consists of potential solutions, named particles. These particles are randomly initialized and move across the multi-dimensional search space to find the best position according to an optimization function. During optimization, each particle adjusts its trajectory through the problem space based on the information about its previous best performance (personal best, *pbest*) and the best previous performance of its neighbors (global best, *gbest*). Eventually, all particles will gather around the point with the highest objective value.

The position of individual particles is updated as follows:

$$x_i^{t+1} = x_i^t + v_i^{t+1} \quad (7)$$

With v, the velocity calculated as follows:

$$v_{id}^{t+1} = w \times v_{id}^t + c_1 \times r_1 \times (pbest_{id}^t - x_{id}^t) + c_2 \times r_2 \times (gbest^t - x_{id}^t) \quad (8)$$

Where v_i^t indicates velocity of particle i at iteration t, w indicates the inertia factor, C_1 and C_2 indicate the cognition and social learning rates, which determine the relative influence of the social and cognition components. r_1 and r_2 are uniformly distributed random numbers between 0 and 1, x_i^t is current position of particle i at iteration t, $pbest_i^t$ indicates best of particle i at iteration t, $gbest^t$ indicates the best of the group. The algorithm is depicted in Algorithm 1.

Algorithm 1 PSO

Input: termination condition; particle update parameters; fitness function *f*
Initialize particles with random position & velocity
repeat
  foreach particle i
    if $f(pbest_i) <= f(x_i)$
      $pbest_i = x_i$
    end if
  end foreach
  set *gbest* as best *pbest*
  foreach particle i
    update $velocity_i$ and $position_i$
  end foreach
until *termination condition*
output *gbest*

The output of the algorithm is the best global position (solution) found during all iterations. Even though PSO convergence to a global optimum has not been proven for the general case, the algorithm has been shown efficient for many optimization problem. Moreover, PSO has already been applied in classification problem to obtain optimal relevant parameters of traditional classification model, so as to

improve the performance of standard classifier methods. Most of it concerns rule-based classifiers, for instance, PSO is used to extract induction rules to classify data (Sousa, T., Silva, A., & Neves, A., 2004). The standard particle swarm optimizer (PSO) and adaptive Michigan PSO (AMPSO) are applied to the prototype selection problem, and the experimental results show they imrpove the results of the Nearest Neighbor classifiers (Cervantes, A., Galván,. O.M., & Isasi, P., 2009). PSO can also be employed to compute the weights for combining multiple neural network classifiers (Nabavi-Kerizi, S.H., Abadi, M., & Kabir, E., 2010). Additionally, some study demonstrates the feasibility of applying an existing Particle Swarm Optimization approach to feature selection for filtering the irrelevant attributes of the dataset, resulting in a fine Bayesian network built with the K2 algorithm (Chávez, M.C., Casas, G., Falcón, R., Moreira, J.E., & Grau, R., 2007).

MEASURE OPTIMIED COST SENSITIVE LEARNING

Measure optimized cost sensitive learning framework

In this section, we present a new measure optimized framework for optimizing the cost sensitive learning (MOCSL), which uses a Particle Swarm Intelligence to carry out the meta-learning, then we introduce the algorithm procedure of MOCS-SVM and MOCS-NN.

Since the evaluation measures describe the overall performance of classifier, it is more appropriate to evaluate and train the classifier as a whole. As we know, SVM and neural network are both driven by error based objective functions. SVM tries to minimize the regularized hinge loss; neural network tries to minimize the square error. We have known the overall accuracy is not appropriate evaluation measure for imbalanced data classification. As a result, there is an inevitable gap between the evaluation measure by which the classifier is to be evaluated and the objective function according to the classifier is trained. The classifier for imbalanced data learning is needed to be driven by the more appropriate measures. We inject the appropriate measures into the objective function of the classifier in the training with PSO. The common evaluation for imbalanced data classification is G-mean and ROC (Receiver Operating Characteristic) curves. However, for many classifiers, the learning process is still driven by error based objective functions. This paper explicitly treat the measure itself as the objective function when training the cost sensitive learning for improve the performance of classifiers and discovering the best parameter and feature subset. We designed a measure optimized training framework for dealing with imbalanced data classification issue. Chalwa (Chawla, Cieslak, Hall & Joshi, 2008) proposed a wrapper paradigm that discovers the amount of re-sampling for a data set based on optimizing evaluation functions like the F-measure, AUC. To date, there is no research about training the cost sensitive classifier with measure based objective functions. This is one important issue of hindering the performance of cost-sensitive learning.

Another important issue of applying the cost-sensitive learning algorithm to the imbalanced data is that the cost matrix is often unavailable for a problem domain. The misclassification cost plays a crucial role in the construction of cost sensitive approach, and the knowledge of misclassification costs is urgently required for achieving expected classification result. For binary class classification, the cost parameter (ratio misclassification cost) is only one parameter which means the relative cost information, and the cost information to be optimized is only for regulating the accuracy of two classes. However, the values of costs are commonly given by domain experts, it often keep unknown in many domain where it is in fact difficult to specify the precise cost ratio information. It is not exact to set the cost ratio to the inverse of the imbalance ratio (the number of majority instances divided by the number of minority instances); especially it is not accurate for some classifier such as SVM.

Apart from the ratio misclassification cost information, feature subset selection and the intrinsic parameters of the classifier have a significant bearing on the performance. The both of two factors are not only important for imbalanced data classification, but also for any classification. Feature selection is the technique of selecting a subset of discriminative features for building robust learning models by removing most irrelevant and redundant features from the data. Optimal feature selection can concurrently achieve good accuracy and dimensionality reduction. Unfortunately, the imbalanced data distribution are often

accompanied by the high dimensional in real-world data sets such as text classification and bioinformatics. It is important to select features that can capture the high skew in the class distribution (Lusa, 2013; Van Hulse, Khoshgoftaar, Napolitano & Wald, 2009; Zheng, Wu & Srihari, 2004). Moreover, proper intrinsic parameter setting of classifiers, such as regularization cost parameter and the kernel function parameter for SVM, as well as the structure parameters (i.e. number of hidden layers and their nodes) for neural network, can improve the classification performance. For example, for SVM, it is common to use the grid search to optimize the regulation parameter and the kernel parameter. Moreover, these three factors influence each other. Therefore, obtaining the optimal ratio misclassification cost, feature subset and intrinsic parameters must occur simultaneously.

Based on the reason above, our specific goal is to devise a strategy to automatically determine the optimal factors during training of the cost sensitive classifier oriented by the imbalanced evaluation criteria (G-mean and AUC). It is a wrapper framework for empirically discovering the potential misclassification cost ratio, feature subset, and intrinsic parameters for cost sensitive learning (CSL).

In this paper, for the multivariable optimization, especially the hybrid multivariable, the best methods are swarm intelligence technique. We choose the particle swarm optimization (PSO) as our optimization method due to its fast and effective solution space exploration. In addition, many experiments claim that PSO has equal effectiveness but superior efficiency over the GA (Hassan, Cohanim & De Weck, 2005).

Because feature is discrete and parameters are continuous, and the variable needed to be optimized are enormous and mixed. The PSO is a good solution for hybrid multi-variables to be utilized.
The PSO was originally developed for continuous valued spaces; however, many problems have in addition features defined for discrete valued spaces where the domain of the variables is finite. We need to combine the discrete and continuous values in the solution representation since the costs and parameters we intend to optimize are continuous while the feature selection is discrete. Each feature is represented by a 1 or 0 for whether it is selected or not. The major difference between the discrete PSO (Khanesar, Teshnehlab & Shoorehdeli, 2007) and the original version is that the velocities of the particles are rather defined in terms of probabilities that a bit will change to one. Using this definition a velocity must be restricted within the range [0, 1], to which all continuous values of velocity are mapped by a sigmoid function:

$$v''_i = sig(v_i^t) = \frac{1}{1+e^{-v_i^t}} \tag{9}$$

Equation 9 is used to update the velocity vector of the particle while the new position of the particle is obtained using Equation 10.

$$x_i^{t+1} = \begin{cases} 1 & if \quad r_i < v''_i \\ 0 & otherwise \end{cases} \tag{10}$$

Where r_i is a uniform random number in the range [0,1].

Many methods are proposed to deal with the issue of the imbalanced data classification by means of swarm intelligence. Chawla et al implement a genetic algorithm based framework to weight the contribution of each classifier by an appropriate fitness function, such that the classifiers that complement each other on the unbalanced dataset are preferred, resulting in significantly improved performances (Chawla & Sylvester, 2007). Yuan et al proposed to train the standard AdaBoost on training sets oversampled by SMOTE and, in an offline mode, retrain the weights of base classifiers assigned by the standard AdaBoost using Genetic Algorithms (GAs) with G-mean as the fitness function to boost the performance of AdaBoost on imbalanced datasets (Yuan & Ma, 2013). Both of the above methods also belong to wrapper methods of optimizing some parameters. In addition, Yu et al proposed ACOSampling that is a novel undersampling method based on the idea of ant colony optimization (ACO) to address this problem (Yu, Ni & Zhao, 2013). Gao et al a powerful technique for two-class imbalanced classification problems by combining the synthetic minority over-sampling technique (SMOTE) and the particle swarm optimisation (PSO) aided radial basis function (RBF) classifier (Gao, Hong, Chen & Harris, 2011).

Evaluation metrics

Evaluation measures play a crucial role in both assessing the classification performance and guiding the classifier modeling. The purpose of cost-sensitive learning is usually to build a model with total minimum misclassification costs. However, it should be based on the known cost matrix condition. In this article, the purpose of our cost sensitive learning is to get a best AUC or G-mean evaluation metric. And we train the cost sensitive learning using performance measures as the objective functions directly. Through training the cost sensitive classifier with measure based objective functions, we can discover the best factors in terms of the different evaluation. For imbalanced datasets, the evaluation metric should take into account the imbalance. The average accuracy is not an appropriate evaluation metric. We used the G-mean and AUC to evaluate the cost sensitive classifiers. The evaluation metrics value is taken as the fitness function to adjust the position of a particle. These two different evaluations reflect different aspect of the classifier. The AUC concerns the ranking ability more and the G-mean concerns the two accuracies of both classes at the same time.

The G-mean is the geometric mean of specificity and sensitivity, which is commonly utilized when performance of both classes is concerned and expected to be high simultaneously (Kubat & Matwin, 1997). It is a good indicator on overall performance, and has been used by several researchers for evaluating classifiers on imbalanced datasets (Akbani, Kwek & Japkowicz, 2004; Barua, Monirul Islam, Yao & Murase, 2013). According to the confusion matrix mentioned in the section 1, we define the sensitivity, specificity and G-mean as follows:

$$Sensitivity = \frac{TP}{TP + FN} \quad Specificity = \frac{TN}{TN + FP} \qquad (11)$$

$$G - mean = \sqrt{Sensitivity * Specificity} \qquad (12)$$

ROC analysis (abbr. of Receiver Operating Characteristic) has been recently introduced to evaluate machine learning algorithm. ROC curves measure the separating ability of a classier between two classes. It depicts all possible trade-off between TP rate and FP rate. Closely related to ROC, AUC represents a ROC curve as a single scalar value by estimating the area under the curve, varying between 0 and 1. The AUC measures the performance of ranking a randomly chosen positive example higher than a randomly chosen negative example. In this case, it represents the performance of ranking an instance from the minority class higher than instances in the majority class. The value 1of AUC represents all positives being ranked higher than all negatives. The authors in (Ling, Huang, & Zhang, 2003) have empirically and formally prove that AUC is a statistically consistent and more discriminating measure than accuracy. It is also as the measure criteria for evaluating performance of classification on the imbalanced dataset (Chawla, Cieslak, Hall & Joshi, 2008; Klement, Wilk, Michaowski & Matwin, 2009). Since AUC is believed to be a better performance measure than accuracy for imbalanced classification problems, and independent of class prevalence. Many existing learning algorithms been modified to deal with the new objective (Tang, Wang & Chen, 2011).

MOCS-SVM

The solution (i.e. particle) of MOCS-SVM includes three parts: the ratio misclassification cost C_{rf}, the intrinsic parameters (C and γ) of classifier, and the feature subsets. Figure 1 illustrates the mixed solution representation in the PSO. If n features are required to decide which features are chosen, then n+3 decision variables must be adopted. The value of n variables of feature ranges between 0 and 1. If the value of a variable is less than or equal to 0.5, then its corresponding feature is not chosen. Conversely, if the value of a variable is greater than 0.5, then its corresponding feature is chosen. In addition to feature selection, three decision variables, C, C_f and γ, are required.

Ratio cost	Intrinsic parameters		Feature subset				
C_{rf}	C	γ	f_1	f_2	...	f_{n-1}	f_n

Figure 1. Solution representation of MOCS-SVM

Figure 2 shows the flowchart for MOCS-SVM. First, the population of particles is initialized, each particle having a random position within the D-dimensional space and a random velocity for each dimension. Second, each particle's fitness for the CS-SVM is evaluated. G-mean or AUC is a criteria used to design a fitness function. Thus, for the particle with high G-mean or AUC produce a high fitness value. The fitness has been taken as the G-mean or AUC. If the fitness is better than the particle's best fitness, then the position vector is saved for the particle. If the particle's fitness is better than the global best fitness, then the position vector is saved for the global best. Finally the particle's velocity and position are updated until the termination condition is satisfied. Associated with the characteristics of exploitation and exploration search, PSO can deal with large search spaces efficiently, and hence has less chance to get local optimal solution than other algorithms.

The detailed algorithm MOCS-SVM to optimize cost sensitive SVM by imbalanced data measure is shown in Algorithm 2. It is a wrapper framework for empirically discovering the potential misclassification cost ratio, feature subset, and intrinsic parameters (C and γ) for CS-SVM oriented by the imbalanced evaluation criteria (G-mean and AUC).

The choice of the fitness function is important because it is on this basis that the PSO evaluates the goodness of each candidate solution for designing our classification system. We employ a 5-fold cross - validation to represent an unbiased estimation of the generalization performance of classifier for each candidate solution. We first split the training data set into five partitions. Each partition is used once as a testing fold, with the remaining 80% as the training fold.

Algorithm 2 MOCS-SVM

Input: Training set D; termination condition T; population size SN; metric E; $NumFolds$ =5
Randomly initialize particle population positions and velocities (including cost matrix, intrinsic parameters, and feature subset)
repeat
  foreach particle i
    Construct the D_i with the feature selected by the particle i
    Separate D_i randomly into $NumFolds$ folds
    for k=1 to $NumFolds$
      Train CS-SVM with cost matrix and intrinsic parameters optimized by the particle i on the k-th training fold Trt^k_i
      Evaluate the cost sensitive classifier on the Trv^k_i, and obtain the value M^k_i based on E
    end for
    M_i=average(M^k_i);Assign the fitness of particle i with M_i
    if $fitness\ (pbest_i) <= fitness\ (x_i)$
      then $pbest_i = x_i$
    end if
  end foreach
  set $gbest$ as best $pbest$
  foreach particle i
    update $velocity_i$ and $position_i$ with Eq. 2 and 3.
  end foreach
until *termination condition*
output optimal parameters, cost ratio and feature subset of $gbest$

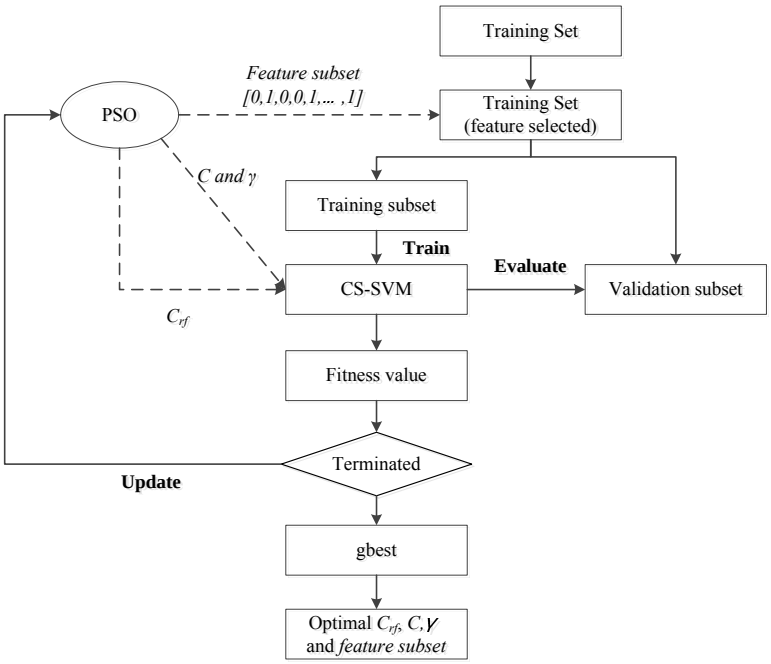

Figure 2. The flowchart of the proposed MOCS-SVM

MOCS-NN

In the training of the feed-forward neural network, it is often trained by adjusting connection weights with gradient descent. Another alternative is to use swarm intelligence to find the optimal set of weights (Yuan & Liu, 2011). Since the gradient descent is a local search method vulnerable to be trapped in local minima, we opted to substitute the gradient descent with PSO in our use of PSOCS-NN in order to alleviate the curse of local optima. We use a hybrid PSO algorithm similar to the PSO-PSO method presented in (Carvalho & Ludermir, 2007). In the PSO-PSO Methodology, a PSO algorithm is used to search for architectures and a PSO with weight decay (PSO: WD) is used to search for weights. We also used two nested PSOs, where the outer PSO is used to search for architectures (including the feature subset which determines the input node amount as well as the number of the hidden nodes) and miscalssification costs; the inner PSO is used to search for weights of the neural network defined by the outer PSO. The procedure of inner PSO is the same as the method proposed in (Mazurowski, M.A., Habas, P.A., Zurada, J.M., Lo, J.Y., Baker, J.A. & Tourassi, G.D. 2008), and the major motivation of using evolutionary techniques over gradient based learning algorithms for training neural networks is to alleviate the curse of local optima. We assume there is only one hidden layer. The solution of the outer PSO includes three parts: the cost, the number of the hidden nodes and the feature subset, and the solution of the inner PSO contains the vector of the connection weights. The amount of the variables to be optimized in the inner PSO is determined by the number of the hidden nodes in the outer PSO. Figure 3 illustrates the mixed solution representation of the two PSOs. Figure 4 shows the flowchart for MOCS-NN. The detailed algorithm for MOCS-NN is shown in Algorithm 3.

Figure 3. Solution representation of MOCS-NN

Outer PSO	Ratio cost	number of the hidden nodes	Feature subset				
	C_{rf}	N	f_1	f_2	...	f_{n-1}	f_n
Inner PSO	*Weight vector*						
	w_1	w_2	...		w_{N-1}		w_N

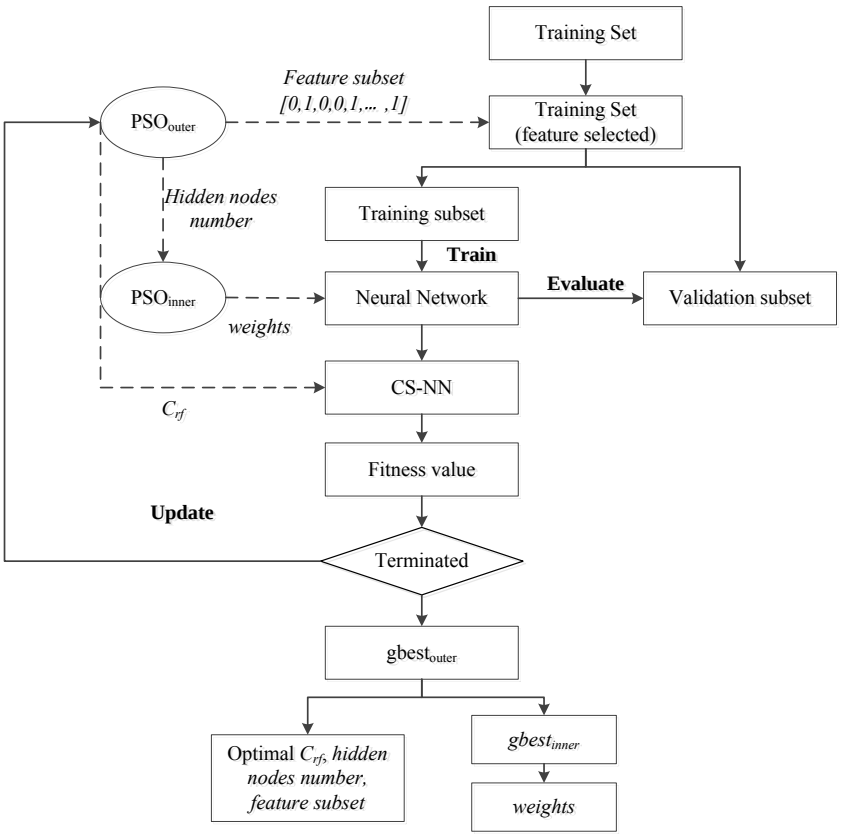

Figure 4. The flowchart of the proposed MOCS-NN

Algorithm 3 MOCS-NN

Input: Training set D; Termination condition of two PSO T_{outer} and T_{inner};
Population size of two PSOs SN_{outer} and SN_{inner}
Randomly initialize outer-PSO population (including costs, number of the hidden nodes, and feature subset)
repeat *% outer PSO*
foreach particle^i
Construct D^i with the feature selected by the particle^i
Separate D^i randomly into Trt^i (80%) for training *and* Trv^i (20%) for validation
Randomly initialize inner-PSO population (connection weights) in each particle^i
repeat *% inner PSO*
foreach particle$^i j$
Obtain the number of the hidden nodes from the particle^i
Construct a neural network with the weights optimized by the particle$^i j$
Validate the neural network on the Trt^i and assign the fitness of particle$^i j$ with the G-mean
end foreach
Inner-PSO particle population updates
until T_{inner}
Obtain the optimal connection weight vector in the $gbest^i{}_{inner}$ of the inner PSO
Evaluate the neural network classifier with cost optimized by the particle^i as well as
the connection weights optimized on the Trv^i, and obtain the value M^i based on G-mean

Assign the fitness of particle^i with M^i
end foreach
Outer-PSO particle population updates
until T_{outer}
Output: the number of the hidden nodes, costs, feature subset and the connection weights of the $gbest_{inner}$

EXPERIMENTAL STUDY

Dataset description

To evaluate the classification performance of our proposed methods in different classification tasks, and to compare with other methods specifically devised for imbalanced data, we tried several datasets from the UCI database. There is no standard dataset for imbalanced classification, and most of these selected UCI datasets have multi-class labels. We used all available datasets from the combined sets used in (Akbani, Kwek & Japkowicz, 2004). This also ensures that we did not choose only the datasets on which our method performs better. We also keep the same minority class as the paper (Akbani, Kwek & Japkowicz, 2004) using one class as the positive class (minority), while the union of all others as the negative class. The minority class label (+) is indicated in Table 2. The datasets chosen have diversity in the number of attributes and imbalance ratio. Moreover, the datasets used have both continuous and categorical attributes.

We first split the data set into ten partitions. Each partition is used once as a testing fold, with the remaining 90% as the training fold. This results in ten pairs of training and testing folds (10-fold cross-validation), and to prevent overtraining, the training set is separated into training subset (80%) for constructing the classification model and test subset (20%) for evaluating and calculating the fitness value in each fold. The training and validation sets were characterized by the same ratio of both class. I made vertical comparison and horizontal comparison. The vertical comparison means the comparison between our method proposed and the intermediate method or basic method, such as basic classifier, cost sensitive learning and grid search optimization for CS-SVM. The horizontal comparison is the comparison between our method MOCSL and the state-of the-art methods for class imbalance learning.

Table 2. The data sets used for experimentation
The dataset name is appended with the label of the minority class (+)

Dataset (+)	Instances	Features	Class balance
Hepatitis (1)	155	19	1:4
Glass (7)	214	9	1:6
Segment (1)	2310	19	1:6
Anneal (5)	898	38	1:12
Soybean (12)	683	35	1:15
Sick (2)	3772	29	1:15
Car (3)	1728	6	1:24
Letter (26)	20000	16	1:26
Hypothyroid(3)	3772	29	1:39
Abalone (19)	4177	8	1:130

Experiment 1 (vertical comparison): how the MOCSL improves

In the vertical comparison, we made the comparison between basic classifier with and without the feature selection, cost sensitive learning (CSL), our method proposed using measure oriented training for CSL by PSO (MOCSL) with and without the feature selection. For SVM, we also apply the common grid search optimization method for comparison. For the basic classifier with feature selection, it is a common wrapper feature selection method with evaluating by classification performance. As for the CSL, the misclassification cost ratio is search iteratively for maximize the measure score within a range of cost

value. As for the optimizing the CS-SVM using grid search, we also need to treat this misclassification cost ratio as a hyperparameter, and locally optimize this parameter. However, it is not feasible to use a triple circulation for optimizing the best parameters, so we optimize the best parameter pair(C and γ), then locally optimize the cost ratio parameter based on the best parameter pair(C and γ) before. All SVMs model in this experiment use the same kernel, RBF, and for basic SVM and CS-SVM, the intrinsic parameters are chosen with default values (C=1 and γ =1). In the basic neural network and CS-NN, the number of neurons in the hidden layer was the average number between the input and output neurons.

For the PSO setting of our method, MOCSL, the initial parameter values of it in our proposed method were set according to the conclusion drawn in (Carlisle, 2001). The parameters were used: C_1=2.8, C_2=1.3, w=0.5. For empirically providing good performance while at the same time keeping the time complexity feasible, particle number was set dynamically according to the amount of the variables optimized (=1.5×|variables need to be optimized|), and the termination condition could be a certain number of iterations (500 cycles) or other convergence condition (no changes any more within 2× |variables need to be optimized| cycles).

Along with these parameters in PSO, the other parameters are the upper and lower of limit parameter of model to be optimized. For SVM, the ranges for C and γ are based on a grid search for SVM parameters as recommended in (Hsu, Chang & Lin, 2003). The range of C is (2^{-5}, 2^{15}), and the range of C is (2^{-15}, 2^{3}). For the neural network, the upper and lower limits of the connection weights were set to 100 and -100 respectively in the inner PSO; the upper and lower limits of the hidden node amount were empirically set to 5 and 20 respectively in the outer PSO. The range of ratio misclassification cost factor C_r was empirically chosen between 1 and 100×*ImbaRatio* (the ratio between the instance amounts of two classes).

Table 3. Experimental results (AUC) of the MOCSL method with and without feature selection, as well as basic method and grid search for SVM

	SVM						**Neural network**				
Dataset	Basic		CSL	Grid-CSL	MOCSL		Basic		CSL	MOCSL	
	without *FS*	*FS*	without *FS*	without *FS*	without *FS*	*FS*	without *FS*	*FS*	without *FS*	without *FS*	*FS*
Hepatitis	0.632	0.714	0.707	0.801	**0.861**	0.855	0.851	0.847	0.855	0.859	**0.877**
Glass	0.952	0.957	0.953	0.955	0.994	**1**	0.932	0.945	0.956	0.987	**0.994**
Segment	1	1	1	1	1	1	0.999	1	**1**	**1**	**1**
Anneal	0.876	0.925	0.957	**1**	**1**	**1**	0.886	0.898	0.888	0.909	**0.932**
Soybean	1	1	1	1	1	1	**1**	1	**1**	**1**	**1**
Sick	0.728	0.761	0.788	0.848	0.908	**0.975**	0.817	0.823	0.862	0.924	0.941
Car	0.990	0.987	0.990	0.999	**1**	**1**	0.996	0.996	0.998	**1**	**1**
Letter	0.898	0.895	0.909	0.983	0.980	**0.999**	0.955	0.962	0.972	0.979	**1**
Hypothyrid	0.830	0.855	0.887	0.945	0.973	**0.988**	0.951	0.963	0.963	0.968	**0.972**
Abalone	0.638	0.712	0.722	0.839	0.867	0.893	0.851	0.853	0.884	**0.891**	0.875

In this experiment, we assess the overall quality of classifiers with only the AUC evaluation metric. The average AUC scores are shown in the Table 3. From the result in Table 3, we found that simultaneously optimizing the feature subset, parameter and cost ratio generally help the base classifiers learned on the different data sets, regardless of feature selecting or not.

For the classifier with the default model intrinsic parameters, the neural network is better than SVM, it is because that SVM is much more sensitive to the choice of model intrinsic parameter than neural network. The default model parameters of SVM cannot get the best performance.

Meanwhile, for SVM, under the condition where the feature selection is not carried out, we found the optimization for all the factors simultaneously using PSO outperform the optimization using extent grid search, which optimize the intrinsic parameter firstly, then search the optimal misclassification cost parameter based on the best intrinsic parameters. It lacks sufficient search in the parameter space, many potential parameters pairs not to be abtained in the parameter space. Hence, it shows that the parameters need to be search at the same time. We believe that a wrapper method can allow one to empirically

discover the relevant parameters, as it is certainly intrinsically tied in with the data properties. Many results are particularly suggestive because they show that the degree of imbalance is not the only factor that hinders learning. Consider the Anneal data set, which is certainly not the most unbalanced data set. Both SVM and neural network have a very poor performanc. However, there is a significant improvement offered by the wrapper methods.

We also found the feature selection step for these classifiers when working on the imbalanced data classification for both the basic classifier and the MOCSL. In the MOCSL, the use of feature selection was found to improve the AUC on the most datasets.

We have demonstrated the ability to optimize the parameters and input of classifier model for an evaluation function, resulting in effective generalization performance.Therefore, we can draw the conclusion that the simultaneously optimizing the intrinsic, misclassification cost parameter and feature selection with the imbalanced evaluation measure guiding improve the classification performance of the cost sensitive learning on the different datasets. Moreover, the average AUC score of MOCS-SVM is better than MOCS-NN, which demonstrate that our wrapper approach improve the SVM much more than neural network.

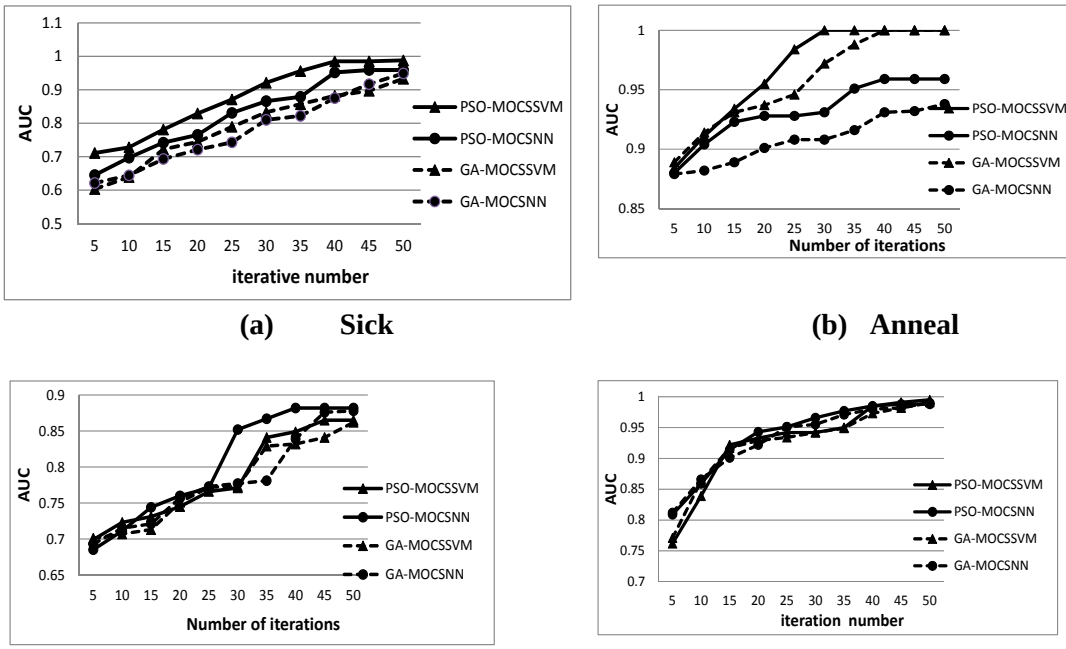

Figure 5. The comparison between GA and PSO in terms of AUC and number of iterations

Genetic algorithms also have the potential to generate both the optimal feature subset and parameters at the same time. Figure 5 shows a graphical evolution in terms of the average of AUC of execution of PSO based wrapper and GA based wrapper on four datasets. Both the methods performs feature selection and parameters setting in an evolutionary way. Compared with GA method, PSO based approach generally achieve higher AUC. Moreover, we can also see that for a given AUC value, PSO based wrapper methods tends to take fewer iterations to converge.

Compared with GAs, PSO does not need complex operators such as crossover and mutation, it requires only primitive and simple mathematical operators, hence it is computationally inexpensive in terms of both memory and runtime

Experiment 2 (horizontal comparison): MOCSL vs. the state-of the-art methods

The horizontal comparison means the comparison that our method and the other state-of-the-art imbalanced data classifiers, such as the random under-sampling (RUS), SMOTE over-sampling (Chawla, Bowyer, Hall & Kegelmeyer, 2002), SMOTEBoost (Chawla, Lazarevic, Hall & Bowyer, 2003) and SMOTE combined with asymmetric cost classifier (Akbani, Kwek & Japkowicz, 2004). For the under-sampling algorithm, the SMOTE and SMOTEboost, the re-sampling rate is unknown. The common method for RUS is that majority class of the training data is randomly under-sampled until the sizes of both classes are the same. The common method for SMOTE is that the minority class was oversampled at the different rates from 100% to 500% and choose the average of these different results of different ratio oversampling as the final result. In our experiments, in order to compare equally, no matter under-sampling or over-sampling method, we also use the evaluation measure as the optimization objective of the re-sampling method to search the optimal re-sampling level. The increment step and the decrement step are both set as the 10%. This is a greedy search, which process repeats, greedily, until no performance gains are observed. The optimal re-sampling rate is decided in a greedy iterative fashion according to the evaluation metrics. Thus, in each fold, the training set is separated into training subset and validating subset for searching the appropriate rate parameters. The evaluation metrics are also used with the G-mean and AUC. For the SMOTE with asymmetric cost classifier, for each re-sampling rate searched, the optimal misclassification cost ratio is determined by grid search under the evaluation measure guiding under the current over-sampling level of SMOTE. Any algorithm that tries to improve on it inevitably sacrifices some specificity in order to improve the sensitivity. G-mean metric is the best of the three measures because it combines both the sensitivity and the specificity and takes their geometric mean.

The experiment results are shown in the Table 4 and Table 5. As shown in bold in Table 4 and 5, our MOCSL outperforms all the other approaches on the great majority of datasets. Irrespective of the wrapper evaluation function, the wrapper approaches always result in an improved G-mean and AUC over the base classifier. For MOCSL based on the SVM (MOCS-SVM), it did not get the best result only on the Glass dataset; For MOCSL based on the neural network (MOCS-NN), there are only two dataset (Soybean and Hypothyroid) not to be winner. From the results, we can see that the random under-sampling is with worst performance. This is because that it is possible to remove certain significant examples. Especially for SVM, undersampling the majority class causes larger angles between the ideal and learned hyperplane.

Both the SMOTE and SMOTEBoost generally help the base classifiers learned on the different data sets. Using the SMOTE based technique of oversampling the minority instances, we can make the distribution of positive instances denser. SMOTE or SMB synthetically generates new instances between two existing positive instances which helps in making their distribution more well-defined. However, SMOTE itself makes some assumptions about the training set. For instance, it assumes that the space between two positive instances is assumed to be positive and the neighborhood of a positive instance is also assumed to be positive, which may not always be true. Since our algorithm uses SMOTE, it also makes a similar assumption. In some complex datasets where this assumption may not hold, such as Hepatitis and Sick，our algorithm will perform slightly worse than the other algorithms.

The over-sampling algorithm that tries to improve on it inevitably sacrifices some specificity in order to improve the sensitivity; but the degree of sensitivity improved is larger than the one of specificity improved. However, they have a potential disadvantage of distorting the class distribution. SMOTE combined with different cost classifier is better than single only SMOTE over-sampling, and it is the method that share most of the second best results. For some dataset, such as Segment, Soybean and Car, the AUC can be achieved 1, which indicates perfect ranking performance,and the two classes can be differenated easily.

There is not a distinct positive correlation between the objective functions in the wrapper-mode and corresponding improvements in the final evaluation. In majority the cases, the G-mean value from the G-mean wrapper is higher than the one of the AUC wrapper, but in some cases, the G-mean value from the AUC wrapper is higher, such as Hepatitis and Abalone datasets for MOCS-SVM and Glass and letter

datasets for MOCS-NN. Even for MOCS-SVM, the average G-mean from AUC optimization is better than the one from G-mean optimization. From this, we believe these results in more generalized performances when using AUC as the wrapper evaluation function, which is the similar conclusion as the paper (Chawla, Cieslak, Hall & Joshi, 2008), where the F-measure on some data sets when using AUC as the wrapper evaluation metric rather than the F-measure. We believe that employing the AUC evaluation measure as optimization objective could lead to more generalized performances. Similarly, the two evaluation metrics wrapper optimization for the same classifier result in different misclassification cost, feature subset and intrinsic parameters, since that they optimize different properties of the classifier.

Table 4. Experimental comparison between MOCSL method and other imbalanced data classification methods based on the SVM

Dataset	**Metric**	**RUS**		**SMOTE**		**SMB**		**SMOTE-CSL**		**MOCSL**	
		AUC	**GM**	**AUC**	**GM**	**AUC**	**GM**	**AUC**	**GM**	**AUC**	**GM**
		Optimization metric		Optimization metric		Optimization metric		Optimization metric		Optimization metric	
Hepatitis	**AUC**	0.663	0.528	0.754	0.721	0.788	0.759	0.813	0.783	**0.855**	0.823
	GM	0.598	0.487	0.672	0.667	0.558	0.592	0.628	0.729	**0.805**	0.801
	Fea	19								7	8
Glass	**AUC**	0.955	0.948	0.988	0.986	0.981	0.978	0.992	0.975	**1**	0.995
	GM	0.817	0.803	0.844	0.858	0.874	0.862	0.965	**0.988**	0.986	0.971
	Fea	9								5	4
Segment	**AUC**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**
	GM	0.993	**1**	**1**	**1**	**1**	**1**	**1**	**1**	0.998	**1**
	Fea	19								10	11
Anneal	**AUC**	0.882	0.866	0.912	0.876	0.891	0.889	0.957	0.934	**1**	**1**
	GM	0.616	0.535	0.758	0.821	0.761	0.784	0.819	0.835	0.999	**1**
	Fea	38								14	12
Soybean	**AUC**	**1**	0.992	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**
	GM	0.876	0.953	0.947	0.965	0.992	0.997	**1**	0.997	**1**	**1**
	Feature	35								12	12
Sick	**AUC**	0.784	0.742	0.822	0.799	0.841	0.824	0.931	0.874	**0.975**	0.954
	GM	0.206	0.141	0.452	0.528	0.508	0.512	0.811	0.825	0.893	**0.915**
	Feature	29								9	7
Car	**AUC**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**
	GM	0.964	0.964	0.962	0.958	0.979	0.981	0.995	**0.998**	0.996	**0.998**
	Feature	6								4	4
Letter	**AUC**	0.907	0.896	0.966	0.956	0.987	0.965	0.988	0.980	**0.999**	0.995
	GM	0.925	0.933	0.947	0.954	0.934	0.922	0.965	0.961	0.983	**0.985**
	Fea	16								12	10
Hypothyroid	**AUC**	0.876	0.843	0.971	0.915	0.967	0.955	0.973	0.971	0.988	**0.989**
	GM	0.482	0.612	0.853	0.894	0.876	0.903	0.876	0.901	0.964	**0.968**
	Fea	29								9	14
Abalone	**AUC**	0.781	0.613	0.822	0.754	0.799	0.780	0.846	0.812	**0.893**	0.855
	GM	0.618	0.687	0.712	0.814	0.645	0.744	0.698	0.817	**0.853**	0.785
	Fea	8								4	5

Table 5. Experimental comparison between MOCSL method and other imbalanced data classification methods based on the neural network

Dataset	Metric	RUS		SMOTE		SMB		SMOTE-CSL		MOCSL	
		AUC	GM	AUC	GM	AUC	GM	AUC	GM	AUC	GM
		Optimization metric		Optimization metric		Optimization metric		Optimization metric		Optimization metric	
Hepatitis	**AUC**	0.751	0.611	0.795	0.74	0.823	0.815	0.841	0.827	**0.893**	0.877
	GM	0.756	0.793	0.829	0.835	0.812	0.807	0.822	**0.851**	0.832	0.848
	Fea	19								10	9
Glass	**AUC**	0.932	0.919	0.985	0.964	0.987	0.988	0.975	0.953	0.987	**0.994**
	GM	0.845	0.847	0.841	0.851	0.843	0.885	0.931	0.965	0.963	**0.970**
	Fea	9								5	5
Segment	**AUC**	**1**	0.999	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**
	GM	0.996	0.993	0.998	0.999	0.995	0.998	0.999	**1**	0.998	**1**
	Fea	19								14	14
Anneal	**AUC**	0.919	0.902	0.884	0.856	0.878	0.839	0.911	0.847	**0.932**	**0.932**
	GM	0.676	0.702	0.766	0.799	0.797	0.848	0.861	0.914	0.907	**0.934**
	Fea	38								18	16
Soybean	**AUC**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**
	GM	0.862	0.948	0.988	**1**	**1**	**1**	0.997	**1**	0.999	**1**
	Fea	35								14	15
Sick	**AUC**	0.768	0.721	0.843	0.817	0.853	0.856	**0.965**	0.885	0.962	0.941
	GM	0.325	0.354	0.682	0.699	0.726	0.748	0.822	0.816	0.899	**0.907**
	Fea	29								13	11
Car	**AUC**	0.812	0.806	0.999	0.986	0.998	0.990	**1**	**1**	**1**	**1**
	GM	0.725	0.786	0.923	0.944	0.945	0.939	0.951	**0.988**	0.975	0.969
	Fea	6								4	5
Letter	**AUC**	0.916	0.925	0.958	0.929	**1**	0.943	**1**	0.998	**1**	**1**
	GM	0.943	0.957	0.953	0.959	0.938	0.966	0.972	0.963	**0.978**	0.971
	Fea	16								11	10
Hypothyroid	**AUC**	0.889	0.861	0.944	0.923	**0.979**	0.952	0.944	0.935	0.977	0.972
	GM	0.651	0.673	0.823	0.841	0.842	0.853	0.897	0.917	0.955	**0.958**
	Fea	29								12	13
Abalone	**AUC**	0.797	0.751	0.811	0.793	0.804	0.771	0.837	0.828	**0.888**	0.875
	GM	0.644	0.726	0.733	0.748	0.741	0.756	0.828	0.857	0.823	**0.856**
	Fea	8								5	6

The feature selection is as important as the re-sampling in the imbalanced data classification, especially on the high dimensional datasets. However, the feature selection is always ignored. Our method conduct the feature selection in the wrapper paradigm, hence improve the classification performance on the data sets which have higher dimensionality, such as Anneal, Sick and Hypothyroid. As expected using different classifiers also results in different feature subset. Because different algorithms have different biases and a feature that may help one algorithm may hurt another, so the feature subsets are different according to different wrapper classifiers. The feature number and feature are both different.

Although all methods are optimized under the evaluation measure optimized, we can see clearly that MOCSL is almost always equal to, or better than other methods. What is most important is that our method does not change the data distribution. The re-sampling based on the SMOTE may make the model overfitting, resulting in the generalization is not as good as the training.

Many papers conclude that there is no consistent clear winner between the sampling approaches and the cost-sensitive technique. However, the conclusions were based on the default condition without the sufficient search in the parameters space. In this paper, we have empirically show that the under the evaluation measure guiding, the performances of cost sensitive learning with cost, feature subset and intrinsic parameter optimized are better than the re-sampling methods with sampling level optimized.

Due to the nature of PSO, searching of cost setups might be time-consuming with some applications. This approach is still respectable considering that this searching is usually an off-line procedure such that the learning speed is not a crucial issue.

Experiment 3: Lung computer-aided detection

Many lung nodule computer-aided detection (CAD) methods have been proposed to help expert radiologists in their decision making. A CAD scheme for nodule detection in CT can be broadly divided into two major steps, an initial nodule identification step and a false-positive reduction step (Li, 2007). The purpose of false-positive reduction is to remove these false positives (FPs) as much as possible while retaining a relatively high sensitivity. It is a typical class imbalance issue since that the two classes are skewed and have unequal misclassification costs. The imbalanced data issue usually occurs in computer-aided detection systems since that the healthy class is far better represented than the diseased class in the collected data (Rao, Fung & Krishnapuram, et al, 2009; Yang, Zheng, Siddique, & Beddoe, 2008), including other CAD, such as breast, colony.

Constructing an accurate classification method requires a training data set that represents different aspects of nodule features. As we know, feature extraction plays an important role in computer aided detection. However, there is not a single outstanding feature that can discriminate the nodule from non-nodule completely. This is due to the fact that the nodules vary enormously in volume, shape, and appearance, and the sources of false positives are different. The majority of false positives are mainly caused by blood vessels and other normal anatomic structures. Some of the false positives can be easily distinguished from true nodule, however, a large portion of them are difficult to distinguish. Therefore, for getting a high classification accuracy in candidate nodule classification, we should extract more features from many aspects, such as intensity, shape and gradient. Our feature extraction process generated 43 image features. Using these features, we construct the input space for our classifiers. This section gives a brief introduction to the features we have collected for analysis and selection. Table 6 describes the features extracted from the candidate nodule Volume-Of-Interest (VOI) for classification.

Our database consists of 98 thin section CT scans with 106 solid nodules, obtained from Guangzhou hospital in China. These databases included nodules of different sizes (3-30mm). We obtained the appropriate candidate nodule samples objectively using a candidate nodule detection algorithm, which identifies 85 true nodules as positive class and 462 non-nodules as negative class from the total CT scans; the class imbalance ratio is 5.4. The imbalance level is not extremely high, but the misclassification costs of each class are extremely different. The imbalance level is dependent on reliability and accuracy of the initial detection processing. The Generation of the nodule candidates is displayed in Figure 6.

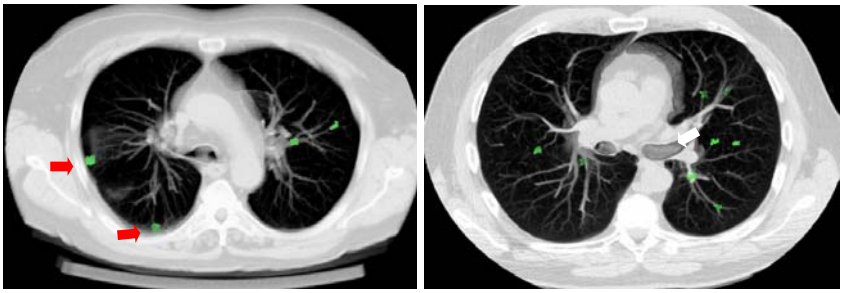

Figure 6. Initial detection result of candidate nodules. TPs indicated by arrow, other spots are FPs

Table 6. Feature set for candidate nodule classification

#	Feature Type	Feature	Description
1-7	Intensity	Intensity statistical feature	The gray value within the objects was characterized by use of seven statistics (mean, variance, max, min, skew, kurt, entropy).
8-12		sub-volume distribution feature	The average intensity within each sub-volume along the radial directions
13-19	Shape	SI statistical feature	The volumetric shape index (SI) representing the local shape feature at each voxel was characterized by use of seven statistics.
20-26		CV statistical feature	The volumetric curvedness (CV), which quantifies how highly curved a surface is, was characterized by use of seven statistics.
27-29		volume, surface area and compactness	
30-36	Gradient	Concentration statistical feature	The concentration characterizing the degree of convergence of the gradient vectors at each voxel, was characterized by use of seven statistics
37-43		Gradient strength statistical feature	The gradient strength of the gradient vectors at each voxel, was characterized by use of seven statistics

Experiments show that the framework proposed improves the evaluation metric, AUC in Table 7. For high dimensional candidate nodule dataset, our methods outperform the other common approach. It means that our method can be applied on the nodule or other lesion detection medical images. The measure optimization is only used the AUC metric.

Table 7. Experiment result of candidate nodule classification

Method	metric	Base	CSL	RUS	SMOTE	SMOTEBoost	SMOTE-CSL	MOCSL
SVM	**AUC**	0.681	0.785	0.603	0.948	0.948	0.956	**0.969**
	GM	0.208	0.662	0.590	0.826	0.818	0.867	**0.937**
NN	**AUC**	0.872	0.899	0.873	0.926	0.925	0.938	**0.946**
	GM	0.513	0.650	0.439	0.858	0.864	0.909	**0.921**

CONCLUSION

Learning with class imbalance is a challenging task. Cost sensitive learning is an important approach without changing the distribution because it takes into account different misclassification costs for false negatives and false positives. Since the cost matrix, the intrinsic algorithm parameters and the feature subset are important factors for the cost sensitive learning, and they influence each other, it is best to

attempt to simultaneously optimize them using an object optimized wrapper approach. We propose a wrapper paradigm optimized by the evaluation measure of imbalanced dataset as objective function with respect to misclassification cost, feature subset and intrinsic parameter of classifier. The optimization processing is through an effective swarm intelligence technique, the Particle Swarm Optimization (PSO). Our measure optimized framework could wrap around an existing cost-sensitive classifier. We demonstrated its applicability with SVM and neural networks, two completely different classifiers. The proposed method has been validated on some benchmark dataset as well as a real world dataset (Lung medical image), which is typically an imbalanced data set with different misclassification cost. The experimental results presented in this study have demonstrated that the proposed framework provided a very competitive solution to other existing state-of-the-arts methods, in optimization of G-mean and AUC for combating imbalanced classification problems. These results confirm the advantages of our approach, showing the promising perspective and new understanding of cost sensitive learning.

FUTURE RESEARCH

Several interesting problems related to this research are still open for future investigation. The following is a list of some possible directions.

(1) More investigations on other base classifier

In this study, we only demonstrated its applicability with SVM as well as neural network which are commonly used in the imbalanced data learning. Other standard classification systems, such as bayesian network classifier, decision tree, and K-NN, are all reported to be affected by the class imbalance problem. Our measure optimized framework can be applied on other classifiers. In future research, we will extend and investigate how the cost sensitive learning wrapper algorithms effect different base classification systems.

(2) More investigations for multiple classes classification

Most existing imbalance learning techniques are only designed for and tested in two-class scenarios. They have been shown to be less effective or even cause a negative effect in dealing with multi-class tasks. In the future research, we will extend the framework to the multiclass imbalanced data classification.

(3) More investigations for other objective function

The setup of optimized parameters is specific not only to the given data, but also to the learning objective and the base classifier. The kind of objective function can be chosen based on the training objective of the given problem; the alternative performance measures such as F-measure can also be incorporated.

REFERENCES

Akbani, R., Kwek, S., & Japkowicz, N. (2004). Applying support vector machines to imbalanced datasets. *Proceedings of the 2004 European conference on machine* learning (pp. 39-50).

Barua, S., Monirul Islam, M., Yao, X., & Murase, K. (2013). MWMOTE: Majority Weighted Minority Oversampling Technique for Imbalanced Data Set Learning. *IEEE Transactions on Knowledge and Data Engineering, In Press*.

Carlisle, A., Dozier, G. (2001). An Off-The-Shelf PSO. *Particle Swarm Optimization Workshop* (pp. 1-6).

Carvalho, M., Ludermir, T.B. (2007). Particle swarm optimization of neural network architectures and weights. *Proceedings of the 7th international conference on hybrid intelligent systems* (pp. 336-339).

Cervantes, A., Galván,. O.M., & Isasi, P. (2009). AMPSO: A New Particle Swarm Method for Nearest Neighborhood Classification. *IEEE Transactions on Systems, Man, And Cyvernetics-Part B: Cybernetics, 39 (5), 1082-1091.*

Chávez, M.C., Casas, G., Falcón, R., Moreira, J.E., & Grau, R. (2007). Building Fine Bayesian Networks Aided by PSO-Based Feature Selection. *Adavances in Artificial Intelligence* (pp. 441-451).

Chawla, N.V., Bowyer, K.W., Hall, L.O., & Kegelmeyer, W.P. (2002). SMOTE: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research,16, 321–357.*

Chawla, N.V., Lazarevic, A., Hall, L.O., & Bowyer, K.W. (2003). SMOTEBoost: Improving Prediction of the Minority Class in Boosting. *Proceedings of the Seventh European conference Principles and Practice of Knowledge Discovery in Databases* (pp. 107-119).

Chawla, N.V., Japkowicz, N., & Kolcz, A. (2004). Editorial: special issue on learning from imbalanced data sets. *SIGKDD Explorations Special Issue on Learning from Imbalanced Datasets*, *6* (*1*), 1-6.

Chawla, N.V., Cieslak, D.A., Hall, L.O., Joshi, A. (2008). Automatically countering imbalance and its empirical relationship to cost. Data Mining and Knowledge Discovery, *17* (*2*), *225-252*.

Chawla, N.V., & Sylvester, J. (2007). Exploiting Diversity in Ensembles: Improving the Performance on Unbalanced Datasets. *Proceedings of the 7th International Workshop on Multiple Classifier Systems* (pp. 397-406).

Chris, S, Taghi, M. K., Jason, V.H., Amri N. (2008). A Comparative Study of Data Sampling and Cost Sensitive Learning. *IEEE International Conference on Data Mining Workshops*.

Galar, M., Fernández, A., Barrenechea, E., & Herrera, F. (2013). Eusboost: Enhancing Ensembles for Highly Imbalanced Data-sets by Evolutionary Undersampling. *Pattern Recognition, In Press*.

Gao, M., Hong, X., Chen, S., Harris, C.J. (2011). A combined SMOTE and PSO based RBF classifier for two-class imbalanced problems. *Neurocomputing*, *74*, *3456-3466*.

Hassan, R., Cohanim, R., De Weck, O. (2005). A comparison of particle swarm optimization and the genetic algorithm. *Proceedings of the 46th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference*. Austin, TX, USA.

He, H., & Garcia, E.A. Learning from imbalanced data. (2009). *IEEE Transactions on Knowledge and Data Engineering*, *21*(*9*), 1263-1284.

He, H., & Ma, Y. (2013). *Imbalanced Learning: Foundations, Algorithms, and Applications*. Wiley-IEEE.

Hsu, C.W, Chang, C.C. & Lin, C.J. (2003). A Practical Guide to Support vector Classification. *National Taiwan University Technical Report*.

Kennedy, J., Eberhart, R.C. (1995). Particle swarm optimization. *Proceedings of IEEE International Conference on Neural Networks* (pp.1942–1948).

Khanesar, M.A., Teshnehlab, M., Shoorehdeli, M.A. (2007). A novel binary particle swarm optimization. *Proceedings of Mediterranean Conference on Control & Automation* (pp. 1–6). Athens, Greece. IEEE.

Klement, W., Wilk, S., Michaowski, W., & Matwin, S. (2009). Dealing with Severely Imbalanced Data. *Workshop on Data Mining When Classes are Imbalanced and Errors Have Costs, PAKDD.*

Kotsiantis, S., Kanellopoulos, D. & Pintelas, P. (2006): Handling imbalanced datasets: A review. *GESTS International Transactions on Computer Science and Engineering*, *30*(*1*), 25-36.

Kubat, M., Matwin, S. (1997). Addressing the Curse of Imbalanced Training Sets: One-Sided Selection. Proceedings of the 14th International Conference on Machine Learning: *Vol. 97*, (pp. 179-186).

Kukar, M., Kononenko, I. (1998). Cost-sensitive learning with neural networks. *European Conference on Artificial Intelligence* (pp.445–449).

Li, Q. (2007). Recent progress in computer-aided diagnosis of lung nodules on thin-section CT. *Computerized medical imaging and graphics, 31, 248-257.*

Li, N., Tsang, I., Zhou, Z. (2012). Efficient Optimization of Performance Measures by Classifier Adaptation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *35*(*6*), 1370-1382.

Ling, C.X., Huang, J., Zhang, H. (2003). AUC: A Statistical Consistent and More Discriminating Measure than Accuracy. *Proceedings of the 18th International Conference on Artificial Intelligence* (pp. 329-341).

Ling, C.X., Sheng, V.S. (2008). Cost-sensitive learning and the class imbalance problem. *Encyclopedia of Machine Learning*, *231-235*.

Lusa, L. (2013). Improved shrunken centroid classifiers for high-dimensional class-imbalanced data. *BMC bioinformatics, 14(1), 64-76.*

Maloof, M.A. (2003). Learning when data sets are imbalanced and when costs are unequal and unknown. *Proceedings of International Conference on Machine Learning Workshop on Learning from Imbalanced Data Sets*.

Martens, D., Baesens, B., Fawcett, T. (2011). Editorial Survey: Swarm Intelligence for Data Mining. *Machine Learning*, *82*(*1*), *1-42*.

Mazurowski, M.A., Habas, P.A., Zurada, J.M., Lo, J.Y., Baker, J.A. & Tourassi, G.D. (2008). Training neural network classifiers for medical decision making: The effects of imbalanced datasets on classification performance. *Neural Networks, 21, 427-436.*

Nabavi-Kerizi, S.H., Abadi, M., & Kabir, E. (2010). A PSO-based weighting method for linear combination of neural networks. *Computers and Electrical Engineering, 36(5), 886-894.*

Sousa, T., Silva, A., & Neves, A. (2004). Particle swarm based data mining algorithms for classification tasks. *Parallel Computing. 30(5) 5/6,767–783.*

Rao, R. B., Fung, G., Krishnapuram, B., Bi, J., Dundar, M., Raykar, V., Yu, S., Krishnan, A., Zhou, X., & Stoeckel, J. (2009). Mining medical images. *Proceedings of the Third Workshop on Data Mining Case Studies and Practice Prize, Fifteenth Annual SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2009)*.

Tang, K., Wang, R., Chen, T. (2011). Towards maximizing the area under the ROC Curve for multi-class classification problems. *Proceedings of the 25th AAAI Conference on Artificial Intelligence* (pp. 483-488).

Van Hulse, J., Khoshgoftaar, T.M., Napolitano, A., & Wald, R. (2009). Feature selection with high dimensional imbalanced data. *Proceedings of* the *9th IEEE International Conference on Data Mining Workshops* (pp. 507–514) Miami, FL, USA.

Veropoulos, K., Campbell, C. & Cristianini, N. (1999). Controlling the sensitivity of support vector machines. *Proceedings of the international joint conference on artificial intelligence* (pp. 55–60).

Weiss G., McCarthy K., Zabar B. (2007). Cost-sensitive learning vs. sampling: Which is best for handling unbalanced classes with unequal error costs? *Proceedings of the 2007 International Conference on Data Mining* (pp. 35-41), CSREA Press.

Yang, Q., Wu, X. (2006). 10 challenging problems in data mining research. *International Journal of Information Technology & Decision Making, 5*(*4*), *597–604*.

Yang, X., Zheng, Y., Siddique, M., & Beddoe, G. (2008). Learning from imbalanced data: a comparative study for colon CAD. *Proceedings of the* Medical Imaging*: Vol.* 6915.

Yu, H., Ni, J., & Zhao, J.(2013). ACOSampling: An ant colony optimization-based undersampling method for classifying imbalanced DNA microarray data. *Neurocomputing*, 101, 309-318.

Yuan, B., & Liu, W.H. (2011). A Measure Oriented Training Scheme for Imbalanced Classification Problems. *Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining Workshop on Biologically Inspired Techniques for Data Mining* (pp. 293–303).

Yuan, B., & Ma, X. (2012). Sampling + Reweighting: Boosting the Performance of AdaBoost on Imbalanced Datasets. *Proceedings of the 2012 International Joint Conference on Neural Networks* (pp. 2680–2685).

Zheng, Z., Wu, X., Srihari, R. (2004). Feature selection for text categorization on imbalanced data. *ACM SIGKDD Explorations*, *6*(*1*), *80-89*.

Zhou, Z.H., Liu, X.Y. (2006). Training Cost-Sensitive Neural Networks with Methods Addressing the Class Imbalance Problem. *IEEE Transactions on Knowledge and Data Engineering*, *18*(*1*), *63-77*.