A PSO-based Cost-Sensitive Neural Network for Imbalanced Data Classification

Peng Cao [1,2], Dazhe Zhao [1] and Osmar Zaiane [2]

[1] Key Laboratory of medical Image Computing of Ministery of Education, Northeastern University, China
[2] Department of Computing Science, University of Alberta, Canada
Cao.p@neusoft.com, zhaodz@neusoft.com, zaiane@cs.ualberta.ca

Abstract. Learning from imbalanced data is an important and common problem. Many methods have been proposed to address and attempt to solve the problem, including sampling and cost-sensitive learning. This paper presents an effective wrapper approach incorporating the evaluation measure directly into the objective function of cost-sensitive neural network to improve the performance of classification, by simultaneously optimizing the best pair of feature subset, intrinsic structure parameters and misclassification costs. The optimization is based on Particle Swarm Optimization. Our designed method can be applied on the binary class and multi-class classification. Experimental results on various standard benchmark datasets show that the proposed method is effective in comparison with commonly used sampling techniques.

Keywords: classification with Class imbalance, Cost-sensitive learning, Neural network, Particle swarm intelligence

1 Introduction

The classification of data with imbalanced data distributions has posed a significant drawback in the performance of the most traditional classification methods, which assume an even distribution of examples among classes [1]. This problem is growing in importance and has been identified as one of the 10 main challenges of Data Mining [2]. 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 [3]. The significant shortcomings with the re-sampling approach are that 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 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 problems.

adfa, p. 1, 2011.

In the cost-sensitive learning, 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 appropriate misclassification costs are unknown. Besides the costs, the feature set and intrinsic parameter of some sophisticated classifiers also influence the classification performance, such as SVM and neural networks. Moreover, these factors influence each other. This is the first challenge. The other is that as we know, for evaluating the performance of cost-sensitive classifier on the skewed data set, the overall accuracy is not sufficient any more. An appropriate evaluation measure is critical in both assessing the classification performance and guiding the classifier in the imbalanced data distribution scenario, such as G-mean and AUC.

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 [4]. The search space is expanded exponentially as the class number increases. Moreover the factors to be searched are mixture including continuous and discrete variables. Therefore, we use Particle Swarm Optimization (PSO) [5] as the optimization strategy due to its fast and effective solution space exploration.

The contributions of this work can be listed as follows:

1) Optimizing the factors (misclassification cost, feature subset and intrinsic structure parameters) simultaneously for improving the performance of cost-sensitive neural network (CS-NN). We use G-mean [6] to guide the optimization of CS-NN.

2) Most existing imbalance data learning so far are still limited to the binary class imbalance problems. There are fewer effective solutions in multi-class imbalance problems, which exist in real world applications. Our method can be applied on the multi-class imbalance data.

2 Proposed Approaches

2.1 Cost-sensitive Neural network

The cost-sensitive learning technique takes misclassification costs into account during the model construction, and does not modify the imbalanced data distribution directly. The standard neural network is cost **in**sensitive. 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.**(1).

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

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.

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 [7]. 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 [8]. 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.**(2).

$$\begin{aligned} C^* &= \underset{C\in\{1,\ldots,M\}}{argmax}(\, p_1^*(C_1 \mid x),\ldots,\, p_M^*(C_M \mid x)) \\ &= \underset{C\in\{1,\ldots,M\}}{argmax}(\eta_1 cost(C_1)\, p(C_1 \mid x),\ldots,\eta_M cost(C_M)\times p(C_M \mid x)) \end{aligned} \tag{2}$$

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. η_i is a normalization term such that $\sum_{i=1}^{M} P_i^* = 1$ and $0 \le P_i^* \le 1$.

2.2 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.

Particle swarm optimization (PSO) is a population-based global stochastic search method attributed to Kennedy and Eberhart to simulate social behavior [4]. 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 than GA [9]. 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:

$$x_i^{t+1} = x_i^t + v_i^{t+1} \tag{3}$$

With *v*, the velocity calculated as follows:

$$v_{id}^{t+1} = w \times v_{id}^t + c_1 \times r_1 \times (pbest_{id}^t - x_{id}^t) + c_2 \times r_2 \times (gbest^t - x_{id}^t) \tag{4}$$

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.

2.3 PSO based cost-sensitive neural network (PSOCS-NN)

In this section, we present a new measure oriented framework for optimizing the cost-sensitive neural network, which uses a particle swarm intelligence to carry out the meta-learning.

An important issue of applying the cost-sensitive learning algorithm to the imbalanced data is that the exact cost parameters are often unavailable for a problem domain. The misclassification cost, especially the ratio misclassification cost, plays a crucial role in the construction of the cost-sensitive approach. It is not correct to set the cost ratio to the inverse of the imbalance ratio (the amount of majority instances divided by the amount of minority instances).

Apart from the misclassification cost information, the intrinsic structure parameters and feature subset selection of the neural network have a significant bearing on the performance. Both factors are not only important for imbalanced data classification, but also for any other classification task. The proper intrinsic structure parameter setting of neural network (i.e. number of hidden layers and connection weights) can improve the classification performance. 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. The imbalanced data distribution are often accompanied by the high dimensional in real-world data sets such as text classification, or bioinformatics. Optimal feature selection can concurrently achieve good performance and dimensionality reduction [3]. Zheng et al[10] suggest that existing measures used for feature selection are not very appropriate for imbalanced data sets. Hulse et al. [11] investigate that the wrapper feature selection is a good approach for imbalanced datasets, which can find potentially interesting feature information not captured by other filter techniques. Furthermore, the feature subset choice influences the appropriate intrinsic structure parameters as well as misclassification costs and vice versa, obtaining these optimal factors of CS-NN 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. In this paper, for the multivariable optimization, especially the hybrid multivariable, the best method is swarm intelligence technique [12]. We choose the particle swarm optimization (PSO) as our optimization method because it is very mature and easy to implement. In addition, many experiments claim that PSO has equal effectiveness but superior efficiency over GA [9]. The wrapper method is called PSOCS-NN, which empirically discovers the potential misclassification costs, the feature subset, and the intrinsic structure parameters for CS-NN.

Evaluation measures play a crucial role in both assessing the classification performance and guiding the classifier modeling. As we known, neural networks are driven by error based objective functions. We have known the overall accuracy is not an 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 trained [13]. The classifier for imbalanced data learning needs to be driven by the more appropriate measures. We inject the appropriate measures, G-mean into the objective function of the classifier in the training with PSO. The G-mean is the geometric mean of accuracies measured separately on each class, which is commonly utilized when performance of each class is concerned and expected to be high simultaneously [13-14]. The value of the evaluation metric is taken as the fitness function to adjust the position of a particle. Through training the CS-NN with G-mean, we can discover the best factors. The G-mean is defined as:

$$G-mean=(\prod_{i}^{M}R_i)^{1/M} \tag{5}$$

where R_i denotes the recall of class C_i, and M is the number of the classes.

In the binary class classification (M=2), given a certain cost matrix, the CS-NN will classify an instance x into positive (+) class if and only if:

$$P(+|x)Cost(+)>P(-|x)Cost(-) \tag{6}$$

Therefore the theoretical threshold for making a decision on classifying instances into positive is obtained as:

$$p(+|x)>\frac{Cost(-)}{Cost(+)+Cost(-)}=\frac{1}{1+C_{rf}} \tag{7}$$

where C_{rf} is ratio of two cost value, C_{rf}= C(+)/C(-). The value of C_{rf} plays a crucial role in the construction of CS-NN for the classification of the binary class data .

Similarly, for the multiple class classification (M>2), we set the cost of the largest class to 1. The other *M-1* ratio cost parameters need to be optimized.

PSO was originally developed for continuous valued spaces; however, the feature set is discrete, each feature is represented by a 1 or 0 for whether it is selected or not. 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. The major difference between the discrete PSO [15] 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:

$$v'^{t}_{i}=sig(v_i^t)=\frac{1}{1+e^{-v_i^t}} \tag{8}$$

Equation 8 is used to update the velocity vector of the particle while the new position of the particle is obtained using Equation 9.

$$x_i^{t+1}=\begin{cases}1 & if \quad r_i<v'^{t}_{i}\\ 0 & otherwise\end{cases} \tag{9}$$

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

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 [13]. 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 [16]. 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 costs; the inner PSO is used to search for weights of the neural network defined by the outer PSO. In our work, 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 1** illustrates the mixed solution representation of the two PSOs. The detailed algorithm for PSOCS-NN is shown in Algorithm 1.

Algorithm 1 PSOCS-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 $gbest_{inner}$ in the $gbest_{outer}$
```

<table>
<tr><td>Outer PSO</td><td>cost vector</td><td>number of the hidden nodes, N</td><td>F_1</td><td>F_2</td><td>…</td><td>F_{p-1}</td><td>F_p</td></tr>
<tr><td>Inner PSO</td><td>w_1</td><td>w_2</td><td>…</td><td>w_{N-1}</td><td>w_N</td></tr>
</table>

Fig. 1 Solution representations of outer and inner PSO

3 Experimental study

We present two experiments separately, the binary class imbalanced data and multi-class imbalanced data. These datasets are from the public UCI benchmark [17]. In all our experiments, instead of the traditional 10-fold cross validation which can result in few instances of minority class, each dataset was randomly separated into training set (70%) for constructing classifiers and test sets (30%) for validating the classifiers. This procedure was repeated 20 times for obtaining unbiased results.

3.1 Binary class imbalanced data

3.1.1 Dataset description

To evaluate the classification performance of our proposed methods in different binary class classification tasks, and to compare with other methods specifically devised for imbalanced data, we tried several datasets from the UCI database. We used all available datasets from the combined sets used in [18]. This also ensures that we did not choose only the datasets on which our method performs better. There is no standard dataset for imbalanced classification, and most of these selected UCI datasets have multi-class labels. The minority class label (+) is indicated in **Table 1**.

Table 1 The data sets used for binary imbalanced data classification
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

3.1.2 Experiment 1

In this experiment, we made the comparison between basic neural network (Basic NN) with and without the feature selection, cost-sensitive neural network (CS-NN), our method proposed using G-mean oriented training for CS-NN by PSO (PSO-CSNN) with and without the feature selection. For the Basic NN with feature selection, it is a common wrapper feature selection method with evaluating by classification performance. As for the CS-NN, the misclassification cost ratio is searched iteratively to maximize the measure score within a range of cost value. In the Basic NN and CS-NN, the number of neurons in the hidden layer was the average number between the input and output neurons. They are trained with gradient descent.

For the PSO setting of our method, PSOCS-NN, the initial parameter values in our proposed method were set according to the conclusion drawn in [19]. The parameters used were: 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, the particle number was set dynamically according to the amount of the variables optimized (=1.5×|variables needed 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 needed to be optimized| cycles).

Along with these parameters in PSO, the other parameters are the upper and lower limits of CS-NN parameters to be optimized. For the intrinsic structure parameters of 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 number of hidden nodes were empirically set to 5 and 20 respectively in the outer PSO. The range of $Cost(C_i)$ of each class C_i was empirically chosen to [1, 100×$ImbaRatio_i$], where the $ImbaRatio_i$ is the size ratio between the largest class and each class C_i.

Table 2 Experimental results (average G-mean and size of the feature subset after feature selection)of the PSOCS-NN method with and without feature selection (FS), as well as Basic NN and CS-NN

Dataset	Basic NN		CS-NN	PSOCS-NN	
	without *FS*	*FS*	without *FS*	without *FS*	*FS*
Hepatitis	0.751	0.807 (11)	0.755	0.819	**0.848 (8)**
Glass	0.832	0.845 (5)	0.916	0.957	**0.970 (4)**
Segment	0.993	0.997 (10)	**1**	**1**	**1 (11)**
Anneal	0.736	0.798 (19)	0.818	0.909	**0.934 (12)**
Soybean	0.929	**1 (12)**	**1**	**1**	**1 (12)**
Sick	0.517	0.623 (10)	0.712	0.834	**0.907 (7)**
Car	0.783	0.796 (4)	0.928	0.960	**0.969 (4)**
Letter	0.955	0.962 (9)	0.972	**0.979**	0.971 (10)
Hypothyrid	0.651	0.763 (17)	0.813	0.928	**0.958 (14)**
Abalone	0.751	0.753 (6)	0.784	**0.891**	0.856 (5)

The average G-mean scores and the amount of feature subset are shown in **Table 2**. From the results in **Table 2**, we found that simultaneously optimizing the feature subset, intrinsic structure parameters and costs generally helps the CS-NN learn on the different data sets, regardless of whether there is feature selection or not. We also found the feature selection step for these classifiers when working on the imbalanced data classification for both the Basic NN and the PSOCS-NN. Therefore, we can draw the conclusion that simultaneously optimizing the intrinsic parameters, misclassification costs and feature subset with the imbalanced evaluation measure guiding, improves the classification performance of the cost-sensitive neural network on the different datasets.

3.1.3 Experiment 2

In this experiment, the comparisons are conducted between our method and the other state-of-the-art imbalanced data methods, such as the random under-sampling (RUS), SMOTE over-sampling [20], SMOTEBoost [21], and SMOTE combined with asymmetric cost neural network (SMOTE+CS-NN) [18]. For the re-sampling methods, the re-sampling rate is unknown. In our experiments, in order to compare equally, either under-sampling or over-sampling method, we also use the evaluation measure G-mean as the optimization objective of the re-sampling method to search the optimal re-sampling level. The increment step and the decrement step are set as 50% and 10% separately. This is a greedy search, that repeats, greedily, until no performance gains are observed. Thus, in each fold, the training set is separated into training subset and validating subset for searching the appropriate rate parameters. For the SMOTE+CS-NN, 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.

Table 3 Experimental comparison between PSOCS-NN and other imbalanced data classification methods on the binary imbalanced data

Dataset	**Metric**	**RUS**	**SMOTE**	**SMB**	**SMOTE+CS-NN**	**PSOCS-NN**
Hepatitis	G-mean	0.793	0.835	0.807	**0.851**	0.848
	AUC	0.611	0.74	0.815	0.827	**0.877**
Glass	G-mean	0.847	0.851	0.885	0.965	**0.970**
	AUC	0.919	0.964	0.988	0.953	**0.994**
Segment	G-mean	0.993	0.999	0.998	**1**	**1**
	AUC	0.999	**1**	**1**	**1**	**1**
Anneal	G-mean	0.702	0.799	0.848	0.914	**0.934**
	AUC	0.902	0.856	0.839	0.911	**0.932**
Soybean	G-mean	0.948	**1**	**1**	**1**	**1**
	AUC	**1**	**1**	**1**	**1**	**1**
Sick	G-mean	0.354	0.699	0.748	0.816	**0.907**
	AUC	0.721	0.817	0.856	0.885	**0.941**
Car	G-mean	0.786	0.944	0.939	**0.988**	0.969
	AUC	0.806	0.986	0.990	**1**	**1**
Letter	G-mean	0.957	0.959	0.966	0.963	**0.971**
	AUC	0.925	0.929	0.943	0.998	**1**
Hypothyrid	G-mean	0.673	0.841	0.853	0.917	**0.958**
	AUC	0.861	0.923	0.952	0.935	**0.972**
Abalone	G-mean	0.726	0.748	0.756	0.857	**0.856**
	AUC	0.751	0.793	0.771	0.828	**0.875**
Number of Wins / Ties	G-mean	0/0	0/1	0/1	2/2	**6/2**
	AUC	0/1	0/2	0/2	0/2	**7/3**

The experiment results of average G-mean and AUC are shown in **Table 3**. As shown in bold in **Table 3**, our PSOCS-NN outperforms all the other approaches on the great majority of datasets. From the results, we can see that the random under-sampling presents the worst performance. This is because it is possible to remove certain significant examples. Both the SMOTE and SMOTEBoost improve the classification for neural network. However, they have a potential disadvantage of distorting the class distribution. SMOTE combined with different costs is better than single only SMOTE over-sampling, and it is the method that share most of the second best results.

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 conducts the feature selection in the wrapper paradigm, hence improves the classification performance on the data sets which have higher dimensionality, such as Anneal, Sick and Hypothyroid. Although all methods are optimized under the evaluation measure oriented, we can clearly see that PSOCS-NN 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 a weak generalization not as good as the training.

3.2 Multiclass imbalanced data

Most existing imbalance data learning so far are still limited to the binary class imbalance problems. There are fewer solutions in multi-class imbalance problems. They have been shown to be less effective or even cause a negative effect in dealing with multi-class tasks [8]. The experiments in [22] imply that the performance decreases as the number of imbalanced classes increases. We choose six multiclass datasets to evaluate our method. The data information is summarized in **Table 4**. The chosen datasets have diversity in the number of classes and imbalance ratio.

Table 4 The data sets used for multiclass imbalanced data classification

Dataset	Class	Instances	Features	Class distribution
Cmc	3	1473	9	629/333/511
Balance	3	625	4	49/288/288
Nursery	4	12958	8	4320/328/4266/4044
Page	5	5473	10	4913/329/28/88/115
Satimage	6	6435	36	1533/703/1358/626/707/1508
Yeast	10	1484	9	463/429/244/163/51/44/35/30/20/5

We compare our method with the other four methods on the datasets in **Table 4.** The average G-mean values are shown in the **Table 5**. Through the comparison, we found that our method is effective on the multiclass data.

Table 5 Experimental comparison between PSOCS-NN method and other imbalanced data classification methods on the multiclass imbalanced data

Dataset	RUS	SMOTE	SMB	SMOTE+CS-NN	PSO-CSNN
Cmc	0.719	0.741	0.749	0.755	**0.793 (4)**
Balance	0	0.507	**0.562**	0.525	0.542 (3)
Nursery	0.498	0.789	0.811	0.809	**0.853 (4)**
Page	0.684	0.707	0.739	0.758	**0.771 (6)**
Satimage	0.825	0.831	0.841	0.844	**0.872 (15)**
Yeast	0	0.311	0.327	0.335	**0.406 (6)**

4 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 costs, the intrinsic structure parameters and the feature subset are important factors for the cost sensitive neural network, and they influence each other, it is best to attempt to simultaneously optimize them using an object oriented wrapper approach. We propose a wrapper paradigm oriented 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. Our measure oriented framework could wrap around an existing cost-sensitive classifier. 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, on the binary class and multiclass imbalanced data. These results confirm the advantages of our approach, showing the promising perspective and new understanding of cost-sensitive learning.

5 References

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

2 Yang Q, Wu X (2006) 10 challenging problems in data mining research. Int J Inf Technol Decis Mak 5(4):597–604.

3 Chawla NV, 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.

4 Li, N., Tsang, I., Zhou, Z. (2012): Efficient Optimization of Performance Measures by Classifier Adaptation. IEEE Transactions on Pattern Analysis and Machine Intelligence. Volume: PP , Issue: 99, Page(s): 1.

5 Kennedy J, Eberhart RC (1995) Particle swarm optimization, *IEEE Int. Conf. Neural Networks*, pp.1942–1948.

6 Kubat M, Matwin S, et al. (1997) Addressing the curse of imbalanced training sets: one-sided selection, Proc. Int’l Conf. Machine Learning, pp. 179-186.

7 Ling CX, Sheng VS (2008) Cost-sensitive learning and the class imbalance problem, Encyclopedia of Machine Learning Springer.

8 Zhou ZH, Liu XY (2006) Training Cost-Sensitive Neural Networks with Methods Addressing the Class Imbalance Problem. *IEEE Transactions on Knowledge and Data Engineering*, 18(1): pp. 63-77

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

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

11 Van Hulse J, Khoshgoftaar TM, Napolitano A, and Wald R (2009) Feature selection with high dimensional imbalanced data. In*Proceedings of the 9th IEEE International Conference on Data Mining - Workshops (ICDM'09)*:507–514, Miami, FL, December 2009. IEEE Computer Society.

12 Martens D, Baesens B, Fawcett T (2011) Editorial Survey: Swarm Intelligence for Data Mining. Machine Learning, Vol. 82, No. 1, pp. 1-42.

13 Yuan B. & WH Liu (2011) A Measure Oriented Training Scheme for Imbalanced Classification Problems. Pacific-Asia Conference on Knowledge Discovery and Data Mining Workshop on Biologically Inspired Techniques for Data Mining. pp: 293–303.

14 Sun Y, Kamel MS, Wang Y (2006) Boosting for Learning Multiple Classes with Imbalanced Class Distribution. Proc. Int'l Conf. Data Mining: 592-602.

15 Khanesar MA, Teshnehlab M, Shoorehdeli MA (2007) A novel binary particle swarm optimization. In Control & Automation, 2007. MED 07. Mediterranean Conference on, pp. 1–6, Athens.

16 Carvalho M, Ludermir TB (2007) Particle swarm optimization of neural network architectures and weights. In Proc. of the 7th int. conf. on hybrid intelligent systems, pp.336-339.

17 Blake C, Merz C (1998) UCI repository of machine learning databases.

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

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

20 Chawla NV, Bowyer KW, Hall LO, Kegelmeyer WP (2002) SMOTE: Synthetic minority over-sampling technique. J Artif Intell Res 16:321–357.

21 Chawla NV, Lazarevic A, Hall LO, Bowyer KW (2003) SMOTEBoost: Improving Prediction of the Minority Class in Boosting," Proc. Seventh European Conf. Principles and Practice of Knowledge Discovery in Databases, pp. 107-119.

22 Wang S, Yao X (2012) Multiclass imbalance problems: Analysis and potential solutions, Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on 42, pp. 1119-1130.